TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

PAULI MUSTALAHTI

ANTI-SWAY LOAD CONTROL SYSTEM FOR A HYDRAULIC CRANE

Master of Science Thesis

Examiner: Professor Jouni Mattila
Examiner and topic approved in the council meeting of the Engineering Sciences Faculty on 6th April 2016

ABSTRACT

PAULI MUSTALAHTI: Anti-sway control load control system for a hydraulic crane
Tampere University of Technology
Master of Science Thesis, 59 pages, 14 Appendix pages
October 2016
Master's Degree Programme in Automation Technology
Major: Machine automation
Examiner: Professor Jouni Mattila

Keywords: Anti-sway control, LQG, LQR, Kalman filter

The goal of this thesis is to design an anti-sway load control system for a hydraulic crane. In this thesis a tool which is connected to the crane by using two joints is studied. These joints can sway freely in two different directions so the tool includes 2 Degrees-of-freedom (DOF). The test crane in this thesis is Hiab 033 with 3 DOF in the Cartesian space.

At the beginning of this thesis the forward and the differential kinematics equations for the test system are defined. By using these equations, the position of boom tip can be controlled instead of controlling every actuator separately. For the controller design a dynamic model of the tool needs to be determined. In this thesis, the dynamic model is constructed by using Lagrangian dynamic formulation. By comparing the dynamic model with the real system it can be seen that the dynamic model corresponds accurately with the real system.

The linear anti-sway control system is constructed by using the tool dynamic model. The control system includes two main parts. The first controller part is the state feedback controller, which defines reference velocity for the boom tip. From the tool only the swaying angles can be measured so in the control system need to be used Kalman filter needs to be used to estimate missing state velocities. In this thesis state-feedback controller and the Kalman filter is tuned by using Linear-Quadratic-Gaussian (LQG) method. This method combines the Linear-Quadratic-Regulator (LQR) and Kalman filter. The LQR products the optimal tuning matrix for the cost function and the Kalman filter products optimal estimation for the Gaussian white noise. The second control system part is the proportional controllers, which calculate the control signals for the valves according to the reference values.

In this thesis the quality of the anti-sway controller is tested by using two different load mass sizes and two different test paths. According to the results it can be noted that the control system can compensate the tool swaying, when the tool sways forward/backwards. When the tool sways left/right the controller cannot compensate the swaying as efficiently due to the nonlinear flexibility of the swing joint. According to the results it can also be noted that the load mass does not affect the anti-sway load control system.

TIIVISTELMÄ

PAULI MUSTALAHTI: Kuormanvakautusjärjestelmä hydraulista nosturia varten
Tampereen teknillinen yliopisto
Diplomityö, 59 sivua, 14 liitesivua
Lokakuu 2016
Automaatiotekniikan diplomi-insinöörin tutkinto-ohjelma
Pääaine: Koneautomaatio
Tarkastaja: professori Jouni Mattila

Avainsanat: Kuormanvakautus, LQG, LQR, Kalman suodin

Työn tavoitteena on suunnitella kuormanvakautusjärjestelmä hydrauliseen nosturiin. Vakautettavana kuormana tässä työssä käytetään massaa, joka on kiinnitetty hydrauliseen kuormaimeen. Kuormain on liitetty nosturiin käyttämällä kahta vapaasti liikkuvaa niveltä. Nämä nivelet mahdollistavat kouran heilumisen eteen ja taakse sekä sivulle. Testijärjestelmänä tässä työssä käytettään Hiab 033 nosturia, jossa on kolme vapausastetta.

Säätöpiirin suunnittelua varten työssä muodostetaan testijärjestelmälle kinemaattiset yhtälöt. Näiden yhtälöiden avulla on mahdollista ohjata nosturia antamalla ohjausreferenssi puomin pään nopeudelle. Säätimen suunnittelua varten muodostetaan myös dynaaminen malli vakautettavasta kiinnitetystä kuormasta käyttämällä Lagrangen mekaniikkaa. Vertaamalla saatua mallia todelliseen järjestelmään voidaan todeta, että muodostettu malli vastaa hyvin todellista järjestelmää.

Dynaamisen mallin perusteella tässä työssä suunnitellaan lineaarinen säätöjärjestelmä vakauttamaan kuormaa. Tämä säätöpiiri koostuu tilatakaisinkytkennästä, jolla määritetään referenssinopeus puomin kärjelle. Koska kourasta on mahdollista mitata vain heilahduskulmat, säätimessä käytetään Kalman suodinta estimoimaan puuttuvia nopeusmittauksia ja suodattamaan mittaussignaaleja. Tässä työssä tilatakaisinkytkentä ja estimaattori viritetään käyttämällä Linear-Quadratic-Gaussian (LQG) menetelmää, joka yhdistää Linear-Quadratic-Regulator (LQR) menetelmän ja Kalman suotimen. LQR menetelmän avulla voidaan määrittää tiettyä kustannusfunktiota vastaan optimaalisen viritys tilatakaisinkytkennälle. Kalman suodatin puolestaan tuottaa optimaalisen estimaatin LQG ongelmaan. Nopeus referenssin perusteella venttiilien ohjaukset määritetään käyttämällä p-säätimiä jokaiselle puomin toimilaitteelle.

Säätimen toimintaa testataan työssä kahdella erisuuruisella kuormamassalla käyttäen erilaisia liikeratoja. Mittaustulosten perusteella voidaan todeta, että säätöpiiri kykenee kompensoimaan heilurin eteenpäin suuntautuvan värähtelyn tehokkaasti. Heilurin sivuttaista värähtelyä säädin ei kykene yhtä tehokas kompensoimaan nosturin epälineaaristen joustojen vuoksi. Mittaustulosten perusteella voidaan todeta, että kuomamassan kasvattaminen ei vaikuta säätimen toimintaan.

PREFACE

This Master of Science Thesis was carried out at the Department of Intelligent Hydraulics and Automation (IHA) at Tampere University of Technology (TUT).

I would like to express my gratitude to Professor Jouni Mattila for the opportunity to work in IHA and his support and advice. I like also thank Janne Koivumäki and Jyrki Tammisto for their help and advices. Special thanks I would like to address laboratory staff for establish the test system.

First and foremost, I would like to thank my family for the support and encouragement I have received over all these years.

Tampere, 14.10.2016

Pauli Mustalahti

CONTENTS

TERMS AND THEIR DEFINITIONS

Controllability	The system is controllable if and only if it is possible to transfer the system from any initial state to the other desired state by using the unlimited control *u* in limited time
DH-parameter	Denavit-Hartenberg convention parameter
DOF	Degrees of freedom
End-effector	The device, which is connected to the end of the manipulator
Joint variable	The joint variable of a revolution joint is an angle around the rotation axis and the joint variable of a prismatic joint is the joint length
Kalman filter	The estimator which provides the optimal solution to the Linear-Quadratic-Gaussian problem
Lagrangian	Lagrangian represents the difference between the kinetic and potential energy of the system
LQG	Linear-Quadratic-Gaussian control problem where it is assumed that noises of the system are Gaussian white noise
LQR	Linear-Quadratic-Regulator: Optimal steady state tuning gain to the state feedback controller
MIMO	Multi-Input-Multi-Output system
Observability	The system is observable if and only if the initial state of the system can be determined from the early outputs and the control signals in limited time
SISO	Single-Input-Single-Output system

1. INTRODUCTION

Nowadays cranes are used in many different applications like in harbors, in construction sites and in factories to move heavy loads from one place to another. Common to all these places is that there are also workers in same area. For this reason, it is important to move these loads as safely as possible.

Normally, the human operator can command the movements of the crane by controlling every actuator separately and in that way control the boom tip position. When the acceleration of the boom tip changes, the load starts to oscillate because of the mass moment of inertia. The human operator can compensate the swaying of the load by using controls, but this requires a very skilled operator. The anti-sway closed-loop control system could help the operator's work.

The load control systems for a rotary crane and an overhead gantry crane are of considerable research interest. Nevertheless, in literature there are only a few control systems which are experimentally verified for the practical systems. In the past, the load control systems are presented mainly for the gantry cranes (Neupert et al. 2009) and the state of art anti-sway control method for these cranes is called input shaping. In this open-loop control method, the operator control signals are filtered (Ahmad et al. 2009). This filtering removes control components, which cause load oscillation. In practice, these control systems also include a closed-loop feedback controller for the crane position.

For the hydraulic crane many different ways to implement the anti-sway controller in literature are presented for example Palis et al. 2009 and Kim et al. 2004. However, only a few of these control systems are presented for the practical applications. This is done for example in Honkakorpi et al. 2013, Neupert et al. 2009 and Kalmari et al. 2014. One problem in these control systems is how to produce enough feedback information for the controller. In Honkakorpi et al. 2013 MEMS-sensors and in Kalmari et al. 2014 IMU-sensors to produce measurement data are used. Another way to produce data for the control system is to use an estimator (Neupert et al. 2009).

The goal of this thesis is to design an anti-sway control system for a hydraulic crane. The study load is a tool with mass of 100 kg. In this thesis the tool is a common gripper, which can be used to grasp different loads to be used in many applications. For example, the gripper can be used to load logs or biomass. The common way to connect gripper to the crane is used two revolute joints which sway freely in two different direction. The gripper includes rotator but in this thesis rotator is out of scope. For this reason, the gripper includes 2 Degrees-of-freedom (DOF).

In this thesis only the swaying angles of the tool are measured by using incremental encoders, so an estimator needs to be used to reconstruct values for the missing velocity measurements. In this thesis the estimator is tuned by using Linear-Quadratic-Gaussian (LQG) method which produces the optimal estimation for the missing states.

This thesis consists of six chapters. After the introduction, the second chapter presents the mechanical structure of Hiab 033 which is used as the test machine. In the second chapter the forward and differential kinematics equations for the target system are also presented. By using these functions, it is possible to control the Cartesian position of the boom tip instead of common joint control. In the third chapter, the dynamic model for the tool is developed. The fourth chapter presents the theoretical background for the anti-sway load control system design. The control theory is used to design the control system. In the chapter five the test results of the anti-sway controller with two different sizes load masses are presented. The final chapter presents the conclusions of the thesis.

2. KINEMATICS EQUATIONS OF THE CRANE

This chapter presents the construction of the forward and differential kinematics equations for a hydraulic crane with a structure that is very common in material handling and forestry cranes. The forward kinematics equations describe the Cartesian position of the boom tip in a base coordinate frame as a function of the joint variables (joint angles and the extension length). Respectively the differential kinematics equations describe the boom tip Cartesian velocity as a function of the joint velocities. These functions enable the control of the Cartesian position instead of traditional joint control. In this chapter the target system for these equations is Hiab 033.

The construction of the forward and differential kinematics equations requires that the mechanical structure of the crane is known. This crane also has some constant measures, which need to be defined before the construction of the kinematics equations. Therefore, the first section of this chapter presents the basic mechanical structure of Hiab XS 033 B-2. The kinematic equations can be determined by using Denavit-Hartenberg (DH) convention, which is presented in the second section. The final section presents the kinematics equations for the Hiab 033.

2.1 Mechanical structure of Hiab XS 033 B-2

The test system of this thesis is Hiab XS 033 B-2, which is a hydraulic crane with 3DOF in the Cartesian space. This crane is a widely applicable test system for an anti-sway controller because its structure and properties correspond to a very general hydraulic crane. Because the test system is a real crane it is possible to use a realistic load mass in tests. The mechanical structure of the target system and the basic components are depicted in Figure 2.1.

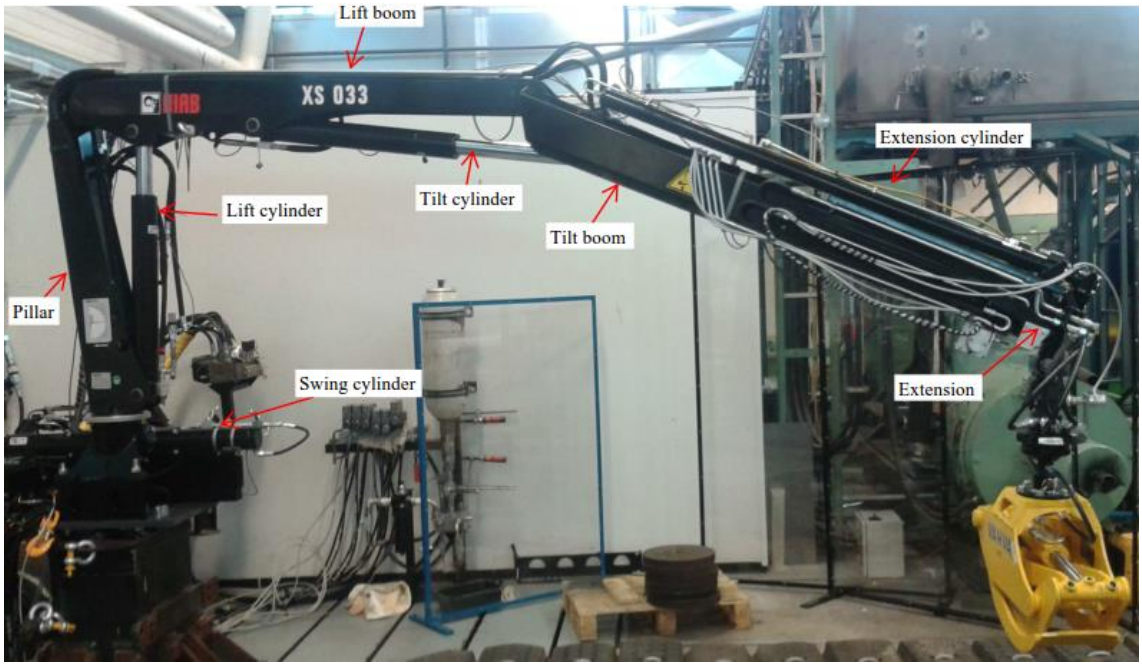

Figure 2.1 *The mechanical structure of Hiab XS 033 B-2*

The crane consists of four booms: a pillar, a lift, a tilt and an extension. All movements of these booms are driven by hydraulic cylinders. All four cylinders are directly operated with NG6 size servo solenoid valves. The nominal flow rates of these valves are 45 l/min (Δ3.5 MPa per control notch). Angle of swing, lift and tilt joints are measured by using incremental encoders and the extension boom length is measured by using a linear position sensor with incremental encoder. These sensors provide accurate joint angle measurements. A supply pressure and the actuators pressures are measured by using pressure sensors. The technical data of these sensors is presented in Appendix A.

The gripper, which is connected at the end of the boom, is Vahva B15 and it is presented more detail in Figure 2.2.

Figure 2.2 *The mechanical structure of the gripper*

The gripper consists of two joints, which can freely sway in two different directions, and the actual gripper part. The total mass of this gripper is about 100 kg. Both tool angles are measured by using incremental encoders. It is also possible to open this gripper by using a hydraulic cylinder and rotate the gripper but in this thesis these features are not used.

The traditional way to implement rotation of the hydraulic crane is to use a pinion gear, which is moved by using hydraulic cylinders. In Hiab, rotation is implemented by using pinion gear which is presented in Figure 2.3.

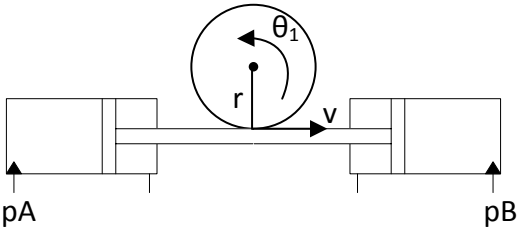

Figure 2.3 *The pinion gear*

The pinion gear consists of the pinion and a rack railway, which is in the cylinders. Cylinder velocity can be calculated by using equation

$$v = r\dot{\theta}_1 \tag{2.1}$$

where v is cylinder velocity and θ is swing joint angle.

2.2 Denavit-Hartenberg convention

A hydraulic crane is a manipulator, which is composed of successive rigid bodies. These bodies are connected to each other by joints. Usually, these joints are either revolute joints or prismatic joints. The revolute joint can rotate around one axis, which is usually *z*-axis in literature. The prismatic joint can move along one axis, which is also *z*-axis in literature. The rigid bodies construct a kinematic chain and an end-effector of the manipulator is connected to the end of this chain (Siciliano et al. 2009, p.58).

Because the structure of the crane is fixed, the end-effector position in a base coordinate frame can be presented as function of the joint variables (Siciliano et al. 2009, p.58). A joint variable of the revolution joint is an angle around the rotation axis and a joint variable of the prismatic joint is the joint length. All manipulator joint variables need to be measured to represent the end-effector position. The end-effector position can be defined by searching a translation and a rotation matrix between the successive coordinate frames. This requires that the coordinate frames of the manipulator are first determined.

The most common way to determine the forward kinematics equations is Denavit-Hartenberg (DH) convention. In this convention the relative orientation and the position between two successive links is defined systematically. According to these properties, a transformation matrix between links can be constructed. When all transformation matrices of the kinematic chain are multiplied by each other, the transformation matrix between the base coordinate frame and the end-effector coordinate frame is obtained (Siciliano et al. 2008, p.26). This transformation matrix can be represented as

$$A_n^0 = A_1^0 A_2^1 A_3^2 .... A_n^{n-1} \tag{2.2}$$

where is A_n^0 the transformation matrix from the base coordinate to the end-effector coordinate frame and other transformation matrices are form

$$A_n^{n-1} = \begin{bmatrix} cv_n & -sv_n c\alpha_n & sv_n s\alpha_n & a_n cv_n \\ sv_n & cv_n c\alpha_n & -cv_n s\alpha_n & a_n sv_n \\ 0 & s\alpha_n & c\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.3}$$

In matrix (2.3) cv_n is *cos* v_n, sv_n is *sin* v_n, $c\alpha_n$ is *cos* α_n, $s\alpha_n$ is *sin* α_n and n is the order number of the link.

In DH-convention only four parameters for each successive coordinate frames need to be defined (Siciliano et al. 2009, p.62). These parameters are presented in Figure 2.4.

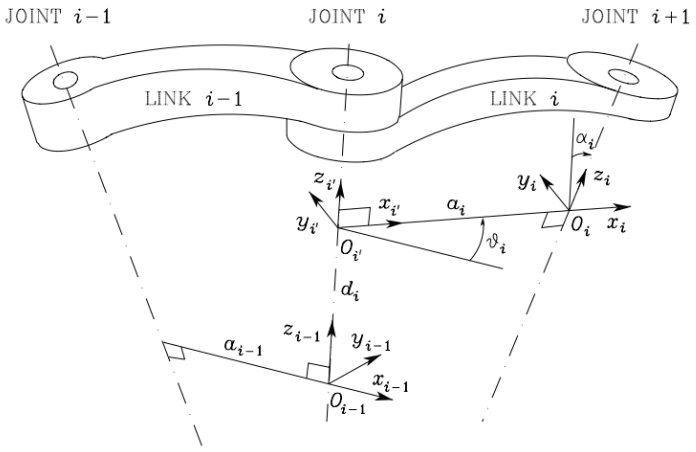

Figure 2.4 *DH-parameters (Siciliano et al. 2009, p.62)*

According to Figure 2.4 the DH-parameters (Siciliano et al. 2008, p.24) are a_i, d_i, α_i and v_i. The parameter a_i is the distance between o_{i-1} and o_i along x_i-axis and the parameter d_i is the coordinate of o_i along z_{i-1}-axis. The parameter v_i is an angle between z_{i-1} and z_i around x_i-axis and the parameter α_i is the angle between x_{i-1} and x_i around z_{i-1}-axis.

These angles are determined so that a positive direction is counter-clockwise and axes are defined by using the right-hand rule.

The DH-convention has some rules to how determine the coordinate frames of the system (Siciliano et al. 2009, p.64). First the z_i-axes of every coordinate frame need to be set at right direction according to type of the joint. Then x_i-axis should be along the common normal vector of axes z_{i-1} and z_i. Finally, y_i-axis can be defined by using right-hand rule.

By usinging these rules, the coordinate frames for the target system can be determined as Figure 2.5 presents.

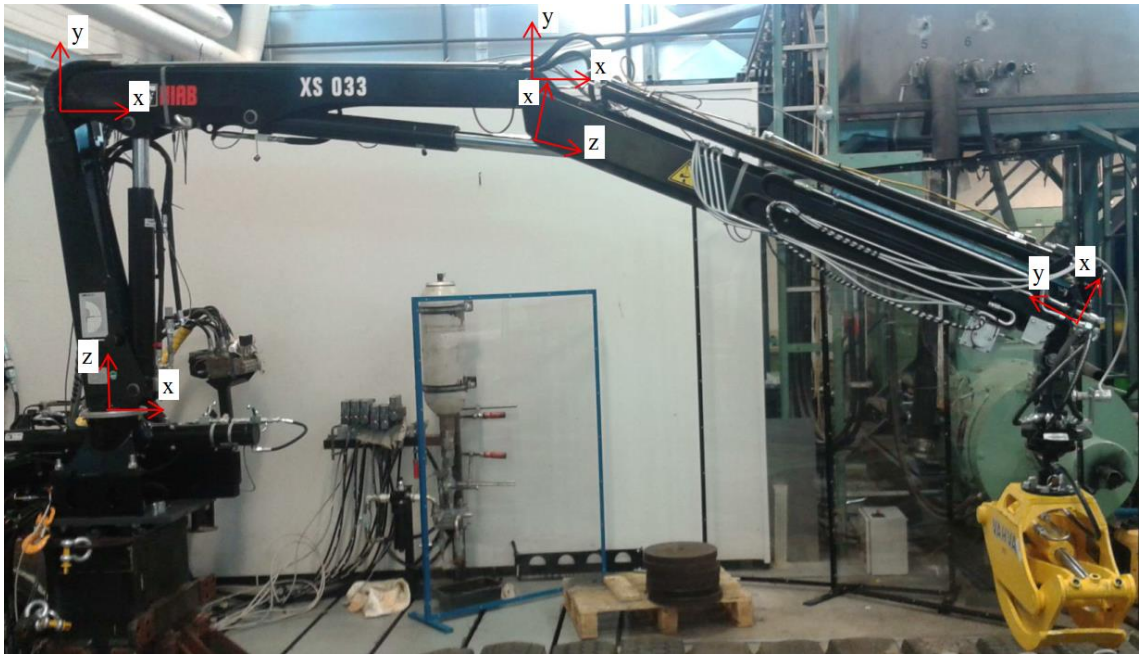

Figure 2.5 *Coordinate frames for DH-convention*

All coordinate frames in Figure 2.5 are selected by using right-hand rule and all angles are also positive counter-clockwise. Now the DH-parameters for all these links can be defined by using parameters in Figure 2.4. The DH-parameters of the test system are presented in Table 2.1.

Table 2.1 *DH-parameters of Hiab XS 033*

Link	*a*	*α*	*d*	*v*
1	o_x	$\pi/2$	o_z	θ_1
2	L_1	0	0	θ_2
3	0	$\pi/2$	L_2	$\pi/2+\theta_3$
4	0	$-\pi/2$	$L_3+\theta_4$	0

In table 2.1 a joint variable θ_1 is the angle of the swing joint, θ_2 is the angle of the lift joint, θ_3 is angle of the tilt joint and θ_4 is the length of the extension boom. The transformation matrix from the base coordinate frame to the end-effector coordinate frame can be constructed by using the transformation matrix in Equation (2.2).

The end-effector position equation can be determined from the last column of the transformation matrix. In this case the end-effector position in the base coordinate frame can be written as

$$\begin{aligned} x &= L_2 \sin(\theta_1) + o_x \cos(\theta_1) + L_1 \cos(\theta_1)\cos(\theta_2) + (L_3 + \theta_4)\cos(\theta_2 + \theta_3)\cos(\theta_1) \\ y &= o_x \sin(\theta_1) - L_2 \cos(\theta_1) + \cos(\theta_2 + \theta_3)\sin(\theta_1)(L_3 + \theta_4) + L_1 \cos(\theta_2)\cos(\theta_1) \\ z &= L_1 \sin(\theta_2) + (L_3 + \theta_4)\sin(\theta_2 + \theta_3) + o_z \end{aligned} \quad . \qquad (2.4)$$

In Equation (2.4) *x*, *y* and *z* are the end-effector coordinates in the base coordinate frame. In Appendix B the symbolic construction of these equations is presented.

The lengths and angles in Table 2.1 and in Equation (2.4) are presented in Figure 2.6.

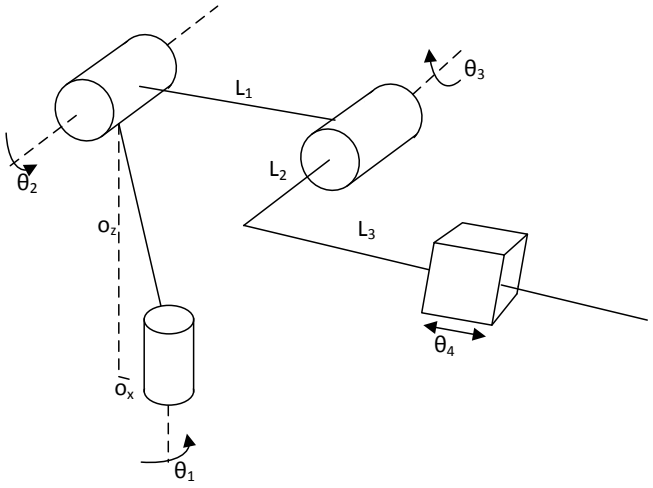

Figure 2.6 *The Manipulator constant angles and lengths*

The constant lengths which affect to DH-parameters are the lift joint offset coordinates o_x and o_z, the lift boom length L_1, the tilt boom length L_3 and L_2 which is the offset between the lift boom and the tilt boom. Other constant lengths of this manipulator is presented in Figure 2.7.

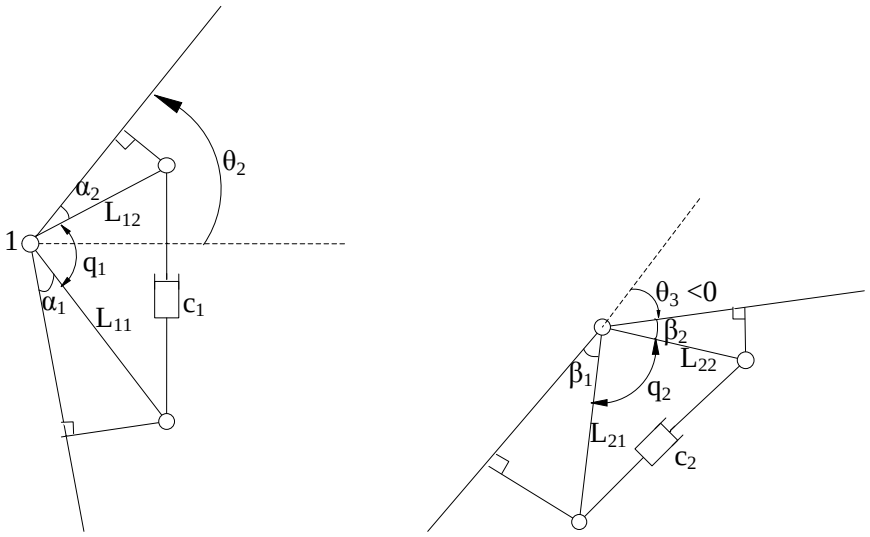

Figure 2.7 *Constant lengths and angles around the cylinders*

The constant lengths are the distances between the joint and the cylinder connection points L_{11}, L_{12}, L_{21} and L_{22}. The constant angles of the crane are angles α_1, α_2, β_1 and β_2.

In Figure 2.7 the joint angle q_1 can be measured and the angle θ_2 can be calculated according to this measurement. By using constant angles and the joint measurement value, θ_2 is (Beiner et al. 1999)

$$\theta_2 = q_1 + \alpha_1 + \alpha_2 - \frac{\pi}{2}. \tag{2.5}$$

Respectively the angle θ_3 can be calculated by using the constant angles β_1 and β_2 and the measured joint angle q_2. So the equation for angle θ_3 is (Beiner et al. 1999)

$$\theta_3 = q_2 + \beta_1 + \beta_2 - \pi. \tag{2.6}$$

Now all lengths and angles in Figure 2.6 and 2.7 are defined.

For a control system the positions of the lift, the tilt and the extension cylinders and the swing joint angle also need to be known. The extension boom length and the swing joint angle can be measured directly from the system, but the lengths of the lift and the tilt cylinders need to be calculated. These lengths can be solved by using the cosine rule (Beiner et al. 1999), which leads to equations

$$\begin{aligned} c_1 &= \sqrt{L_{11}^2 + L_{12}^2 - 2L_{11}L_{12}\cos(q_1)} \\ c_2 &= \sqrt{L_{21}^2 + L_{22}^2 - 2L_{21}L_{22}\cos(q_2)} \end{aligned}. \tag{2.7}$$

Equation (2.7) descripts the actuator lengths as of function of the joint angles.

2.3 Kinematics equations

Equation (2.4) presented the transformation between the joint space and the Cartesian space. Usually this transformation can be written in a more compact (Bajd 2010 et al. p.35) form

$$p = h_1(\theta) \tag{2.8}$$

where p is a position vector

$$p = \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{2.9}$$

The Cartesian velocity and the acceleration can be calculated by differentiating Equation (2.8) twice. The differentiation for these equations results the Jacobian matrix of the joint-to-Cartesian transformation. The Jacobian matrix fulfill equation (Bajd 2010 et al. p.35)

$$\dot{p} = \begin{bmatrix} \frac{\partial x_1}{\partial \theta_1} & \frac{\partial x_1}{\partial \theta_2} & \cdots & \frac{\partial x_1}{\partial \theta_n} \\ \frac{\partial x_2}{\partial \theta_1} & \frac{\partial x_2}{\partial \theta_2} & \cdots & \frac{\partial x_2}{\partial \theta_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial x_m}{\partial \theta_1} & \frac{\partial x_m}{\partial \theta_2} & \cdots & \frac{\partial x_m}{\partial \theta_n} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \vdots \\ \dot{\theta}_n \end{bmatrix} = J(\theta)\dot{\theta} \tag{2.10}$$

By differentiating Equation (2.8) twice relationship between the joint variables accelerations and the end-effector linear accelerations are obtained. This acceleration can be presented as a function (Nakamura 1991, p.20-22)

$$\ddot{p} = J(\theta)\ddot{\theta} + \dot{J}(\theta)\dot{\theta} = J(\theta)\ddot{\theta} + \frac{d}{dt}\frac{\partial J(\theta)}{\partial \theta}\dot{\theta}. \tag{2.11}$$

In Appendix B the symbolic functions for the velocity and the acceleration equations when the system includes 2DOF (Lift and Tilt), 3DOF (Lift, Tilt and Extension) and 4DOF (Swing, Lift, Tilt and Extension) in the actuator space are presented. In Appendix C the symbolic functions for Equation (2.11) when the system includes 2DOF, 3DOF and 4DOF in the actuator space are presented.

Respectively, it is possible to determine the transformation matrices between the joint space and actuator space. This transformation is presented in Equation (2.7) which can be written more compact form

$$c = h_2(q) \tag{2.12}$$

where h_2 is

$$h_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \sqrt{L_{11}^2 + L_{12}^2 - 2L_{11}L_{12}(q_1)} & 0 & 0 \\ 0 & 0 & \sqrt{L_{21}^2 + L_{22}^2 - 2L_{21}L_{22}(q_2)} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.13}$$

The transformations for the swing and extension joints are constant because these joint variables can be measured directly from the real system.

By differentiating Equation (2.12) twice equations for the actuators velocities and accelerations are obtained. The actuators velocities are

$$\dot{c} = A(q)\dot{q} \tag{2.14}$$

and the actuator's accelerations are

$$\ddot{c} = A(q)\ddot{q} + \frac{d}{dt}\frac{\partial A(q)}{\partial q}\dot{q} = A(q)\ddot{q} + \dot{A}(q)\dot{q}\,. \tag{2.15}$$

For Equations (2.14) and (2.15) Jacobian matrix A is

$$A = \begin{bmatrix} r & 0 & 0 & 0 \\ 0 & \dfrac{L_{11}L_{12}\sin q_1}{c_1} & 0 & 0 \\ 0 & 0 & \dfrac{L_{21}L_{22}\sin q_2}{c_2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.16}$$

In Appendix D is presented symbolic functions for Equations (2.12-2.16).

2.4 Inverse kinematics equations

Equation (2.10) presents a ratio between the Cartesian velocity and the joint velocities but usually in robotics the control of the joint velocities according to the Cartesian velocity is wanted. For that reason, the transformation between the joint velocities and the Cartesian velocity needs to be defined. This ratio can be determined from Equation

(2.10) by solving the inverse of the Jacobian matrix (Bajd 2010 et al. p.37) so the transformation between these velocities is

$$\dot{q} = J(q)^{-1}\dot{p}. \tag{2.17}$$

Equation (2.17) is valid only, when the Jacobian matrix is square not redundant. The manipulator is redundant if number of joints is greater than the dimensions of the crane (Nakamura 1991, p.116). When the manipulator is redundant, the Jacobian matrix is not square and it is not possible to solve inverse matrix.

When the manipulator is redundant, the inverse matrix needs to be solved by using a pseudo-inverse solution (Siciliano et al. 2009, p.124). This solution satisfies Equation (2.11) and minimizes a velocities cost function

$$h(\dot{q}) = \frac{1}{2}\dot{q}^T W\dot{q} \tag{2.18}$$

where q is the joint variable vector and W is the $n \times n$ positive definite weighting matrix of the cost function. The optimal solution for Equation (2.11) is (Siciliano et al. 2009, p.125)

$$\dot{q} = W^{-1}J^T(JW^{-1}J^T)^{-1}\dot{p}. \tag{2.19}$$

As shown by Equation (2.19) the pseudo-inverse matrix can be affected by chancing the weight matrix W. For example, a solution which minimizes the kinetic energy can be defined by changing the weighting matrix to manipulator inertia matrix (Beiner et al. 1999). If the weighting matrix W is the unit matrix the solution for the pseudo-inverse

$$J^* = J^T(JJ^T)^{-1} \tag{2.20}$$

where J is the Jacobian matrix. This weighting matrix minimizes the norm of the joint velocities. Respectively the ratio between the joint velocities and the end-effector velocities is

$$\dot{q} = J^*(q)^{-1}\dot{p}. \tag{2.21}$$

In this thesis the Jacobian pseudo-inverse is determined by using Equation (2.21) because it produces enough accuracy for pseudo-inverse solution. In Appendix B the symbolic functions for the inverse matrix when the system includes 2DOF, 3DOF and 4DOF in the actuator space are presented.

3. DYNAMICS EQUATIONS OF THE TOOL

This chapter presents the construction of the dynamic equations for the 2DOF tool. These equations are constructed by using Lagrangian dynamic formulation. The result of this formulation is a nonlinear system model, which can be used in the controller design. Because the model is nonlinear it is not possible to use any linear control methods in the controller design. For this reason, the model needs to be linearized before the controller design.

The first section of this chapter presents the mechanical structure of the tool and the coordinate frames to its joints. The second section presents the construction of the dynamics equations and the final section presents the linearization of these equations.

3.1 Mechanical structure of the tool

In this thesis, the tool is a gripper, which was presented in Figure 2.2. This tool is connected to the crane by using two rigid bodies, which can sway freely in two different directions. In figure 3.1 is presented joints of this gripper.

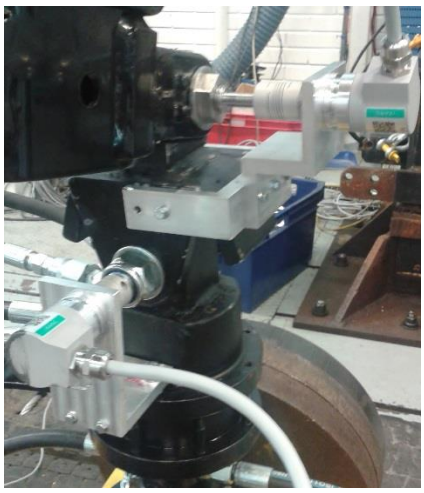

Figure 3.1 *Joints and links of the gripper*

Joints and links in Figure 3.1 can be presented by using simplified mechanical model of the tool, which is presented in Figure 3.2. In the figure is also presented coordinate frames to both joints.

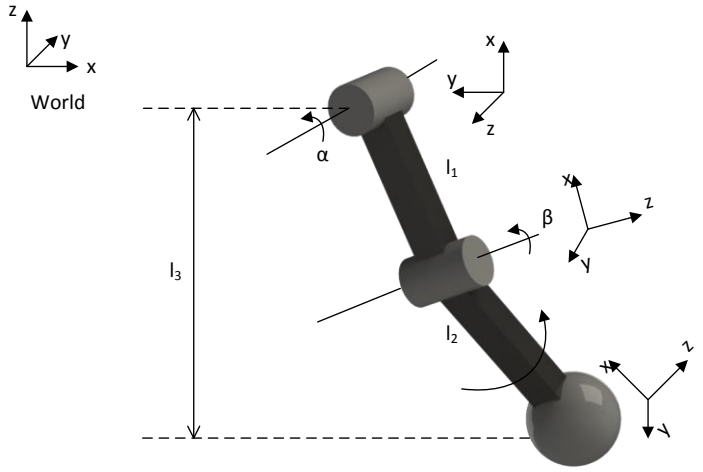

Figure 3.2 *The structure of the tool and the coordinate frames*

Figure 3.2 shows that the tool can also rotate but this property is not used in this thesis. The rigid bodies are connected to each other by using revolute joints and all mass of the tool is assumed to be at the end of the second rigid body. This assumption is valid since l_1 and l_2 weight are only less than 10 kg whereas weight of the gripper is about 90 kg. When the first joint swings the tool sways forward/backward and when the second joint swings the tool sways left/right.

By using Figure 3.2 and the basic trigonometrical functions (Kalmari et al. 2013, p.276) the center of mass position can be defined in the coordinate frame of the end-effector. In this case the position is

$$\begin{aligned} x_m &= -l_2 \sin(\beta) \\ y_m &= -l_3 \sin(\alpha) \\ z_m &= -l_3 \cos(\alpha) \\ l_3 &= l_1 + l_2 \cos(\beta) \end{aligned} \tag{3.1}$$

where x_m, y_m and z_m are the center of mass coordinates and l_1, l_2 and l_3 are lengths of the rigid bodies.

The center of mass position in the world coordinate frame can be determined by using Equation (3.1). According to Figure 3.2 the coordinates in the world coordinate frame are

$$\begin{aligned} x_{cm} &= x_m - l_2 \sin(\beta) \\ y_{cm} &= y_m - l_3 \sin(\alpha) \\ z_{cm} &= z_m + l_3 \cos(\alpha) \\ l_3 &= l_1 + l_2 \cos(\beta) \end{aligned} \tag{3.2}$$

where x_m, y_m and z_m are the coordinates of the boom tip in the world coordinate frame and x_{cm}, y_{cm} and z_{cm} are the center of mass coordinates. The linear velocity and the linear acceleration of the tool can be defined by using Equation (3.2). The linear velocity is obtained by differentiating Equation (3.2) once and the linear acceleration by differentiating the equation twice.

3.2 Lagrangian formulation

One-way to construct the dynamics equations for the tool is to use Newton's laws to solve forces, which affect the tool (Honkakorpi 2013, p.5-6). In this method all force vectors that affect the tool need to be determined, but especially for the complex system the construction of these vectors might be difficult. This is why Lagrangian formulation is a better way to solve the dynamics model (Kalmari et al. 2013 p.276-277).

In Lagrangian formulation the potential and the kinetic energy of the system instead of force vectors to descript model dynamics are used (Jazar 2010, p.620). This method is suitable for the complex system because only the kinetic and potential energy of all joints need to be defined. The kinetic and potential energy do not depend on frictions, which is why they do not appear in these equations. The friction variable effects can be added later to these equations.

At first system Lagrangian need to be determined by using formulation (Jazar 2010, p.620)

$$L = T - V \tag{3.3}$$

where L is Lagrangian, T is the kinetic energy of the tool and V is the potential energy. The Lagrangian represents the difference between system kinetic and potential energy. The kinetic energy of the tool (Freedman et al. 2004, p.214, 340) is

$$T = \frac{1}{2} m v^2 + \frac{1}{2} I \dot{\theta}^2 \tag{3.4}$$

where m is tool mass, v is the tool linear velocity, $\dot{\theta}$ is the tool angular velocity and I is the mass moment inertia. Equation (3.4) shows that the tool kinetic energy consists of the linear and the rotational energy. The total kinetic energy is

$$T = \frac{1}{2} m(\dot{x}_{cm}^2 + \dot{y}_{cm}^2 + \dot{z}_{cm}^2) + \frac{1}{2} I_\alpha \dot{\alpha}^2 + I_\beta \dot{\beta}^2 \tag{3.5}$$

where $\dot{x}_{cm}$, $\dot{y}_{cm}$ and $\dot{z}_{cm}$ are the coordinates linear velocities, $\dot{\alpha}$ and $\dot{\beta}$ are the joints angular velocities and I_a and I_b are the tool mass moment inertias in different direction.

The potential energy depends only on the *z*-coordinate so the potential energy of the tool (Freedman et al. 2004, p.243) is

$$V = mg(z_m - l_3\cos(\alpha)) \tag{3.6}$$

where *m* is the tool mass and *g* is the acceleration of the gravity.

By substituting Equations (3.5) and (3.6) to Equation (3.3) the Lagrangian of the tool is

$$\begin{aligned} L = {} & (\mathrm{m}((\dot{\mathrm{y}} + \dot{\beta}\mathrm{l}_2\cos(\beta))^2 + (\dot{\mathrm{x}} + \dot{\alpha}\cos(\alpha)(\mathrm{l}_1 + \mathrm{l}_2\cos(\beta)) - \\ & \dot{\beta}\mathrm{l}_2\sin(\alpha)\sin(\beta))^2 + (\dot{\mathrm{z}} + \dot{\alpha}\sin(\alpha)(\mathrm{l}_1 + \mathrm{l}_2\cos(\beta)) + \\ & \dot{\beta}\mathrm{l}_2\cos(\alpha)\sin(\beta))^2))/2 + (\mathrm{I}_a\dot{\alpha}^2)/2 + (\mathrm{I}_b\dot{\beta}^2)/2 - \\ & \mathrm{gm}(\mathrm{z} - \cos(\alpha)(\mathrm{l}_1 + \mathrm{l}_2\cos(\beta))) \end{aligned} \tag{3.7}$$

where α is the first joint angle, β is the second joint angle, $\dot{x}$, $\dot{y}$ and $\dot{z}$ are the linear velocities of the boom tip. Equation (3.7) depicts that this method may lead to complex equations. For this reason, it is useful to use symbolic functions to solve these equations. All equations in this chapter are solved by using the symbolic toolbox in Matlab. In Appendix E the symbolic functions for the dynamic model are presented.

When the Lagrangian of the system is defined the joint torques can be solved by using Lagrangian dynamic formulation (Jazar 2010, p.620)

$$\tau_n = \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_n} - \frac{\partial L}{\partial \theta} \tag{3.8}$$

where *n* is the order number of the joint, θ is the joint angle, $\dot{\theta}$ is the joint angular velocity and τ is the torque of the joint. According to Equation (3.8) the torque of the first joint is

$$\begin{aligned} \tau_a = {} & \left(\mathrm{I}_a + \mathrm{l}_1^2\mathrm{m} + \mathrm{ml}_2^2\cos(\beta)^2 + 2\mathrm{ml}_1\mathrm{l}_2\cos(\beta)\right)\ddot{\alpha} + \\ & \left(\mathrm{ml}_1\cos(\alpha) + \mathrm{ml}_2\cos(\alpha)\cos(\beta)\right)\ddot{\mathrm{x}} + \\ & \left(\mathrm{ml}_1\sin(\alpha) + \mathrm{ml}_2\cos(\beta)\sin(\alpha)\right)\ddot{\mathrm{z}} + \\ & \left(\mathrm{ml}_2^2\sin(2\beta) - 2\mathrm{ml}_1\mathrm{l}_2\sin(\beta)\right)\dot{\alpha}\dot{\beta} + \\ & \mathrm{gml}_1\sin(\alpha) + \mathrm{gml}_2\cos(\beta)\sin(\alpha) \end{aligned} \tag{3.9}$$

and the torque of the second joint is

$$\begin{aligned} \tau_\beta = {} & \left(\mathrm{I}_b + \mathrm{l}_2^2\mathrm{m}\right)\ddot{\beta} + \left(-\mathrm{l}_2\mathrm{msin}(\alpha)\sin(\beta)\right)\ddot{\mathrm{x}} + \\ & \left(\mathrm{l}_2\mathrm{mcos}(\beta)\right)\ddot{\mathrm{y}} + \left(\mathrm{l}_2\mathrm{mcos}(\alpha)\sin(\beta)\right)\ddot{\mathrm{z}} + \\ & \left(\mathrm{l}_1\mathrm{l}_2\mathrm{msin}(\beta) + \mathrm{l}_2^2\mathrm{mcos}(\beta)\sin(\beta)\right)\dot{\alpha}^2 + \mathrm{gl}_2\mathrm{mcos}(\alpha)\sin(\beta) \end{aligned} \tag{3.10}$$

From torques (3.9) and (3.10) equations for the angular accelerations $\ddot{\alpha}$ and $\ddot{\beta}$ can be solved by defining the torques which damp the tool swaying. In this case friction, which affects the tool, is modeled by using a viscous friction model (Olsson 1998, p.179). This method does not correspond exactly to the real friction but is accurate enough to describe the friction (Kalmari et al. 2014 p.39). In addition, in this model has only one parameter, which needs to be determined experimentally. The viscous friction model is form of

$$\begin{aligned} \tau_\alpha &= -a_v m\dot{\alpha} \\ \tau_\beta &= -b_v m\dot{\beta} \end{aligned} \tag{3.11}$$

where a_v and b_v are the friction parameters for different joints. The friction parameters need be tuned by comparing the real system with the tool model that is presented in Chapter 5. By adding the friction model to Equations (3.9) and (3.10) the angular acceleration to both joints can be solved. According to Equation (3.9) the linear acceleration of the first joint is

$$\begin{aligned} \ddot{\alpha} = &\left[\left(-l_1 m\cos(\alpha) - l_2 m\cos(\alpha)\cos(\beta)\right)\ddot{x} + \right. \\ &\left(-l_1 m\sin(\alpha) - l_2 m\cos(\beta)\sin(\alpha)\right)\ddot{z} + \\ &-a_v m\dot{\alpha} + \left((l_2{}^2 m\sin(2\beta) + 2l_1 l_2 m\sin(\beta))\right)\dot{\alpha}\dot{\beta} - \\ &\left. gl_1 m\sin(\alpha) - gl_2 m\cos(\beta)\sin(\alpha)\right]/ \\ &(I_a + l_1{}^2 m + l_2{}^2 m\cos(\beta)^2 + 2l_1 l_2 m\cos(\beta)) \end{aligned} \tag{3.12}$$

and respectively the angular acceleration of the second joint is

$$\begin{aligned} \ddot{\beta} = &\left[l_2 m\sin(\alpha)\sin(\beta)\ddot{x} - l_2 m\cos(\beta)\ddot{y} + \right. \\ &\left(-l_2 m\cos(\alpha)\sin(\beta) - l_2 m\cos(\alpha)\sin(\beta)\right)\ddot{z} + \\ &\left.\left(-l_2{}^2 m\cos(\beta)\sin(\beta) - l_1 l_2 m\sin(\beta)\right)\dot{\alpha}^2 - \dot{\beta} b_v m\right]/ \\ &(ml_2{}^2 + I_b) \end{aligned} \tag{3.13}$$

By substituting the Equations (3.12) and (3.13) for the linear acceleration equations, the complete equations to these accelerations are obtained. The tool linear accelerations can be defined by differentiating Equation (3.2) twice. In Appendix F the symbolic differentiating of Equation (3.2) and the substitution of the Equations (3.12) and (3.13) are presented. The linear accelerations of the tool are

$$\ddot{x}_{cm} = \ddot{x} + \cos(\alpha)(l_1 + l_2\cos(\beta))\ddot{\alpha} - \sin(\alpha)(l_1 + l_2\cos(\beta))\dot{\alpha}^2 - l_2\cos(\beta)\sin(\alpha)\dot{\beta}^2 - l_2\sin(\alpha)\sin(\beta)\ddot{\beta} - 2l_2\cos(\alpha)\sin(\beta)\dot{\alpha}\dot{\beta}$$

$$\ddot{y}_{cm} = \ddot{y} + l_2\cos(\beta)\ddot{\beta} - l_2\sin(\beta)\dot{\beta}^2 \quad (3.14)$$

$$\ddot{z}_{cm} = \ddot{z} + \sin(\alpha)(l_1 + l_2\cos(\beta))\ddot{\alpha} + \cos(\alpha)(l_1 + l_2\cos(\beta))\dot{\alpha}^2 + l_2\cos(\alpha)\cos(\beta)\dot{\beta}^2 + l_2\cos(\alpha)\sin(\beta)\ddot{\beta} - 2l_2\sin(\alpha)\sin(\beta)\dot{\alpha}\dot{\beta}$$

By using Equations (3.12 - 3.14) the 2DOF tool dynamics can be described. As equations shows Lagrangian formulation produces the dynamic equations for the complex system quite easily.

3.3 State-space equations and linearization

Dynamics equations (3.12 - 3.14) are nonlinear functions, but because the controller design will be made by using linear control methods, they need to be linearized. The linearized system presents system behavior only around the equilibrium point. When the system distance from the equilibrium point is small, the system acts like linear system. When the system distance is a large, the linearized model does not correspond accurately to the real system because the nonlinearities cause modeling error. In this thesis the linearized system model can be used, because the tool swaying angles can be assumed to be small.

The linear approximation of the system can be defined by constructing Taylor series of the function around the equilibrium point (Khalil 1996, p.37). Taylor series of the function is form

$$F(x) \approx f(x)$$
$$F_n(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2}(x-a)^2 + \ldots + \frac{f^n(a)}{n!}(x-a)^n \quad (3.15)$$
$$= \sum_{k=0}^{n} \frac{f^n(a)}{k!}(x-a)^n$$

where *a* is the operation point and *F(x)* is Taylor series. When the movements around the equilibrium point are small, the high terms of Taylor series are not needed to be taken into account. On the equilibrium point the first term is zero so the linear approximation of function is the second term of Taylor series.

By using linearized functions, the system linear state-space approximation can be defined. The state-space equations for the linear system (Dorf et al. 2005, p.135) are

$$\begin{aligned} \dot{\Delta x}(t) &= A\Delta x(t) + B\Delta u(t) \\ \Delta y(t) &= C\Delta x(t) + D\Delta u(t) \end{aligned} \tag{3.16}$$

where A is the system matrix, B is the control matrix, C is the output matrix and D is the feedforward matrix, Δx is the state vector for the small distances from the equilibrium point, Δy is the output vector for the small distances from the equilibrium point and Δu is the control vector for the small distances from the equilibrium point. By using Equation (3.16) the system structure can be presented as following

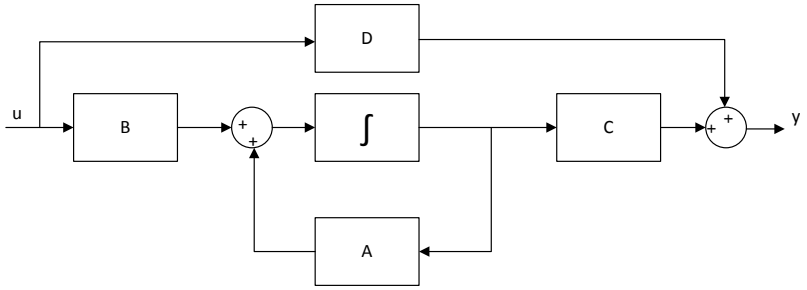

Figure 3.3 *The linear system structure*

In Figure 3.3 u is the control signal and y is output of the system. Usually in the linear system the feedforward matrix D is a zero matrix.

The system equations depend on states and controls so according to Equation (3.15) the state matrix of the linearized system (Khalil 1996, p.37) is

$$A = \begin{bmatrix} \dfrac{\partial f_1(x_e,u_e)}{\partial x_{1e}} & \cdots & \dfrac{\partial f_1(x_e,u_e)}{\partial x_{ne}} \\ \vdots & & \vdots \\ \dfrac{\partial f_n(x_e,u_e)}{\partial x_{1e}} & \cdots & \dfrac{\partial f_n(x_e,u_e)}{\partial x_{ne}} \end{bmatrix} \tag{3.17}$$

where n is the order number of the function, x_e and u_e are the values of the variables in the equilibrium point. The control matrix is respectively

$$B = \begin{bmatrix} \dfrac{\partial f_1(x_e,u_e)}{\partial u_{1e}} & \cdots & \dfrac{\partial f_1(x_e,u_e)}{\partial u_{ne}} \\ \vdots & & \vdots \\ \dfrac{\partial f_n(x_e,u_e)}{\partial u_{1e}} & \cdots & \dfrac{\partial f_n(x_e,u_e)}{\partial u_{ne}} \end{bmatrix}. \tag{3.18}$$

By comparing Equations (3.17) and (3.18) with Equation (2.10) it can be noticed that matrices are Jacobian matrices. For this reason, the linearized matrices can be determined by solving Jacobian matrices. For example, in Matlab the state space matrices

can be defined by using *jacobian*-function. The matrices *C* and *D* can be delineated respectively by using Jacobian matrix.

In Equations (3.12 - 3.14) the state variables are positions, angles and their velocities and the control variables are the linear accelerations of the boom tip. These state variables are chosen because it is possible to solve values for these variables from the real system. The control variables are chosen, because it is possible to control directly these variables. The state variables and the control variables can be written in a compact form

$$\begin{aligned} x &= \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} \end{bmatrix}^T = \\ &\begin{bmatrix} x_{cm} & \dot{x}_{cm} & y_{cm} & \dot{y}_{cm} & z_{cm} & \dot{z}_{cm} & \alpha & \dot{\alpha} & \beta & \dot{\beta} \end{bmatrix}^T \\ u &= \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^T = \begin{bmatrix} \ddot{x} & \ddot{y} & \ddot{z} \end{bmatrix}^T \end{aligned} \tag{3.19}$$

where *x* is the system state vector and *u* is the system control vector.

By using Equations (3.17 - 3.19) the state matrix and the control matrix can be defined on the equilibrium point. In this case in the equilibrium point both joint angles are zero and all velocities are zero. Now the state matrices for the tool are according to Equations (3.17) and (3.18)

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{gm(l_1+l_2)^2}{ml_1^2+2ml_1l_2+ml_2^2+I_a} & -\frac{a_v m(l_1+l_2)}{ml_1^2+2ml_1l_2+ml_2^2+I_a} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{gl_2^2 m}{ml_2^2+I_b} & -\frac{b_v l_2 m}{ml_2^2+I_b} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{-gm(l_1+l_2)}{ml_1^2+2ml_1l_2+ml_2^2+I_a} & \frac{-a_v m}{ml_1^2+2ml_1l_2+ml_2^2+I_a} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-gl_2 m}{ml_2^2+I_b} & \frac{-b_v m}{ml_2^2+I_b} \end{bmatrix} \tag{3.20}$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ \frac{I_a}{ml_1^2+2ml_1l_2+ml_2^2+I_a} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{I_b}{ml_2^2+I_b} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ -\frac{m(l_1+l_2)}{ml_1^2+2ml_1l_2+ml_2^2+I_a} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -\frac{l_2 m}{ml_2^2+I_b} & 0 \end{bmatrix} \tag{3.21}$$

These matrices describe the tool motion near the equilibrium point. The output matrix for this system needs to be defined according to controllability and observability. The defining of the output matrix is presented in Chapter 4. In Appendix F the symbolic construction of Equations (3.20) and (3.21) is presented.

4. CONTROL SYSTEM

This chapter presents a mathematical background for the anti-sway control system. The main parts of this controller are the state feedback controller, Kalman filter, inverse kinematics equations and low-level proportional controllers. The inverse kinematics was already presented in Chapter 2. Because the real-time control interface is based on a PowerPC, the control system is discrete with respect to time, the control system design needs to be done by using the discrete time equations.

The first section presents the discrete time state space model which is used in the controller design. The second section presents the structure of state feedback controller and estimator. The third and the fourth section present Linear-Quadratic-Gaussian (LQG) optimal tuning methods to state feedback controller and estimator. The final section presents the full structure of the anti-sway controller.

4.1 Discrete time state-space model

In Equation (3.16) the linear state space model for the continuous time system was presented. The difference between the continuous time state space model and the discrete time system (Burl 1999 p.16) can be defined as

$$\begin{aligned} A_d &= e^{Ak} \\ B_d &= \int_0^k e^{Fs} ds\, B \end{aligned} \tag{4.1}$$

where k is a sample time of the discrete time system. For example, matrices (3.20) and (3.21) are continuous time so these matrices need to be discretized for the controller design. In Matlab, this transformation is possible to solve by calling *c2d*-function, which solves matrices in Equation (4.1).

The state-space description for the discrete time system can be determined as

$$\begin{aligned} x(k+1) &= A_d x(k) + B_d u(k) \\ y(k+1) &= C_d x(k) + D_d u(k) \end{aligned} \tag{4.2}$$

where A_d is the state matrix, B_d is the control matrix, C_d is the output matrix, D_d is the feedforward matrix and k is the number of sample periods.

4.2 State-feedback control

Usually, the basic feedback controllers cannot produce stable enough results, when the control target is a pendulum. That is why the control system for a pendulum usually includes a state feedback controller. Because the gripper can be modeled as a pendulum, the control system includes the state feedback controller. The state feedback controller is also better suited for controlling a Multi-Input-Multi-Output (MIMO) system (Franklin et al. 1990, p.238) than the basic feedback controllers. The state feedback controller design is usually based on pole placing. This controller sets the poles of the closed-loop system to pre-determined places in the complex state (Dorf et al. 2005, p.666).

The control law for the full-state feedback is

$$u(k) = -Kx(k), \tag{4.3}$$

where u is the control signal vector to the plant, K is a state feedback gain matrix and x is a state vector of the plant. As the control law shows, it assumes that all state variables can be measured. Figure 4.1 presents the block diagram structure of this controller.

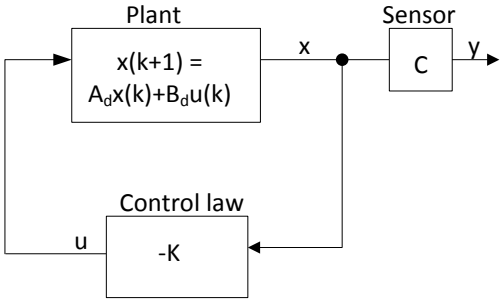

Figure 4.1 *The structure of the full-state feedback controller*

Figure 4.1 depicts that all state variables become directly from the plant to the controller.

In the real applications it is not always possible or wise to measure all state variables. That is why the control system usually includes an estimator, which produces values for the missing state variables. The estimator changes the control law to form

$$u(k) = -K\hat{x}(k) \tag{4.4}$$

where $\hat{x}$ is a state estimation vector. When the system includes, the estimator the structure of the control system is (Franklin et al. 1990, p.265)

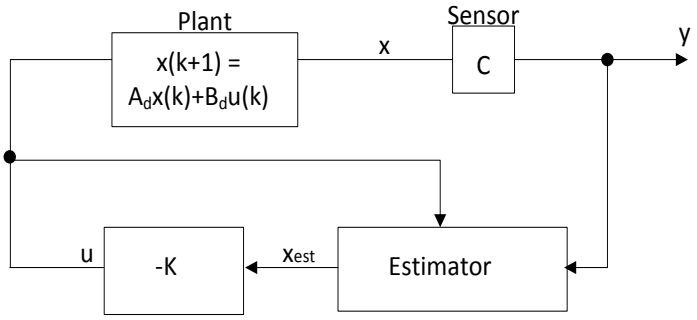

Figure 4.2 *The state feedback controller with the estimator*

Figure 4.2 depicts that in this case state feedback for the controller comes from the estimator. To estimate the missing state variables, the observer needs measurement data from the system and the previous system control signals.

4.2.1 Controllability and observability

As already mentioned, the state feedback controller is based usually on pole-placement techniques. For that all poles of the closed-loop system can be set flexibly, if the system is full controllable and observable (Dorf et al. 2005, p.661). The system is controllable if and only if it is possible to transfer the system from any initial state to the other desired state by using the unlimited control *u* in limited time. The controllability of the MIMO system can be verified by defining the rank of the controllability matrix (Dorf et al. 2005, p.661)

$$rank\begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix} = n \tag{4.5}$$

where the dimensions of the *A* matrix are *n* x *n* and the dimensions of the matrix *B* are *n* x *m*. The system is controllable if and only if the rank of the matrix is equal to *n*. As Equation (4.5) shows the output matrix does not affect controllability.

The system is observable (Dorf et al. 2005 p.661) if and only if the initial state of the system can be defined from the early outputs and the control signals in limited time. The observability of the system can be verified by using matrix

$$Ob = \begin{bmatrix} C \\ CA \\ . \\ . \\ CA^{n-1} \end{bmatrix} \tag{4.6}$$

where C is the output matrix of the system. The system is observable if and only if the rank of this matrix is full.

The controllability and observability of the studied tool model in Equations (3.20) and (3.21) can be verified by using Equations (4.5) and (4.6). According to Equation (4.5) the dynamic model is controllable because the rank of the controllability matrix is

$$rank\begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix} = 10 \tag{4.7}$$

The observability matrix consists of the state matrix and the output matrix. It is not possible to affect the state matrix so the output matrix needs to be defined so that the system is observable. Because the tool has sensors for both joint their angles are a natural choice for the output signals. If the joint angles are only the outputs of the system, the output matrix is

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{4.8}$$

By using Equation (4.6) the rank of the observability matrix is

$$rank\begin{bmatrix} C \\ CA \\ . \\ . \\ CA^{n-1} \end{bmatrix} = 4 \cdot \tag{4.9}$$

Equation (4.9) depicts that the rank of the observability matrix is not full, so only angle measurements are not enough to ensure the observability. That is why the current positions of the center of mass have to be calculated separately by using the joint angles. When these positions are calculated, more numerical values for the state variables are obtained. If the center of mass positions is also measured, the output matrix is

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{4.10}$$

Respectively the rank of the observability matrix is

$$rank\begin{bmatrix} C \\ CA \\ . \\ . \\ CA^{n-1} \end{bmatrix} = 10 \cdot \tag{4.11}$$

As Equation (4.11) depicts the rank of the observability matrix is full so this output matrix ensures the observability of the system. The rank is full also, when only the center of mass coordinates are measured. Because that kind of system cannot be easily implemented, in this case use the output matrix which includes also the swaying angles.

4.2.2 Structure of the estimator

The control law (4.3) assumes that all state variables are available but often this is not possible. That is why the system also includes an estimator to reconstruct the missing state variables. The estimator also offers filtered state values for the measured signals so it is justified to use an estimator, although all state variables are measured.

The most used estimators are a predictor and a current estimator (Franklin et al. 1990, p.250). Both of these estimators calculate a new state estimates by using a previous estimate and previous control signals. The main difference between these two estimators is that the current estimator also uses the state's current values. The predictor uses the previous state values for the estimation. Usually, the current estimator is used more because the control system has no reason to be not use the current measurement information. The predictor is suitable for the control of systems in which a computer cannot calculate everything during one sample period. Practically it is impossible to implement the current estimator, which calculates values with no time elapsed (Franklin et al. 1990, p.257). Instead the current estimator minimizes delays by executing calculations before the next sample.

In this thesis the current estimator is used to reconstruct the missing state variables. The current estimator can be presented by using equation (Franklin et al. 1990, p.257)

$$\hat{x}(k+1) = A\hat{x}(k) + Bu(k) + L(x(k+1) - C(A\hat{x}(k) + Bu(k))) \tag{4.12}$$

where $\hat{x}(k)$ is a previous state estimation vector, $\hat{x}(k+1)$ is a current estimation vector, *x(k+1)* is a current state vector and L is a tuning gain matrix. The structure of the current estimator can be defined by using Equation (4.12). Figure 4.3 presents the current estimator, which is implemented by using basic Matlab blocks.

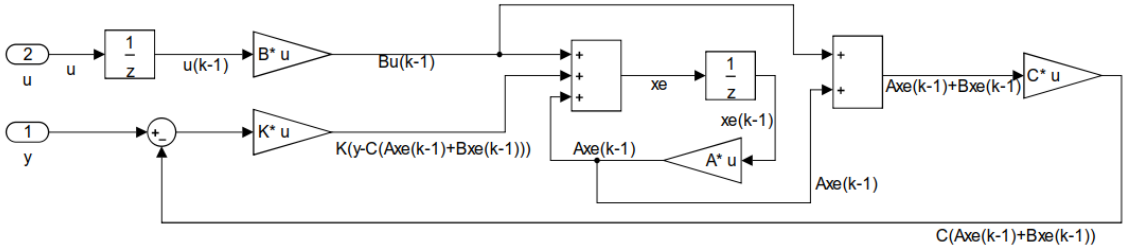

Figure 4.3 *The structure of the current estimator*

As Figure 4.3 depicts that the structure of the estimator is quite simple to implement by using basic matrix calculations.

4.3 Optimal state feedback control

In Figure 4.3 the basic structure of the state feedback controller with the estimator was presented. As shown in Equations (4.3) and (4.11) the state feedback controller and estimator can be tuned with matrices *K* and *L*. If the noises of the system do not correlate to each other (Ostertag 2011, p.260) the state feedback controller and the estimator can be tuned separately according to separation principle (Franklin et al.1990, p.264-266).

The state feedback controller can be tuned by using many different methods. The most common way to tune the controller is use to Ackermann's formulation (Dorf et al. 2005, p.661). According to this formulation, a tuning matrix for the controller is

$$K = \begin{bmatrix} 0 & 0 & \dots & 1 \end{bmatrix} P_c^{-1} q(A) \tag{4.13}$$

where P_c is the controllability matrix and *q(A)* is

$$q(A) = A^n + \lambda_1 A^{n-1} + \dots + \lambda_{n-1} A + \lambda_n I \,. \tag{4.14}$$

In Equation (4.13) gains λ can be defined according to desired characteristic equation

$$q(k) = k^n + \lambda_1 k^{n-1} + \dots + \lambda_n \,. \tag{4.15}$$

As shown in Equations (4.13 – 4.15) the poles of the closed-loop system can be set to the desired locations. In this method has two main problems. First it could be difficult to define the right positions to all poles. The second problem is that in a MIMO system there are many poles to set. These are the main reasons why Ackermann formulation is not easy to use method for MIMO systems.

The more common way to tune the state feedback controller for the MIMO system is to use the optimal control methods. These methods produce the locations for the poles and

it makes possible to determine many candidates for tuning gain matrices (Franklin et al. 1990, p.422). The basic idea of these optimal methods is to solve the pole positions by defining a control, which minimizes the cost function

$$J = \sum_{k=0}^{N} x(k)^T Q x(k) + u(k)^T R u(k) \tag{4.16}$$

where Q is the weighting matrix for the states and R is the weighting matrix for the control signals. This function minimizes the sum of squares of these matrices. By changing the weighting matrices, the designer can affect the pole positions. The tuning matrix, which minimizes the cost function, is optimal solution for the state feedback controller with desired weighting matrices.

It is possible to define the control gain matrix K in such a way that it depends on time (Franklin et al. 1990, p.430) but usually a constant gain matrix is sufficient. The constant gain can be determined by studying the steady state problem in infinite time. The steady state gain is usually sufficient, because the system is used for a long time and an early transient does not affect next transients. This steady state solution is called Linear Quadratic Regulator (LQR), because the system is linear and the cost functions are quadratic.

The steady-state gain, which minimizes Equation (4.16), can be solved by using Riccati equation (Franklin et al. 1990, p.430)

$$A_d{}^T S A_d - S - A_d{}^T S B_d (R + B_d{}^T S B_d)^{-1} B_d{}^T S A_d + Q = 0 \tag{4.17}$$

where S is the solution for Riccati equation. The solution for Equation (4.17) is not unique so the correct answer for the matrix S needs to be chosen. The right solution for Riccati equation should be a positive and a semidefinite matrix so the right solution can be solved. When this matrix is substituted to equation (Franklin et al. 1990, p.438)

$$K = (B_d{}^T S B_d + R)^{-1} (B_d{}^T S A_d) \tag{4.18}$$

the gain matrix for the state feedback controller is obtained. In Matlab, the state feedback controller can be defined by using *dlgr*-command, which solves Equations (4.17) and (4.18).

Equations (4.17) and (4.18) can be used to minimize a quadratic cost function, when the disturbances do not affect the system. In real applications the noise affects measurement values so the disturbances need to be noticed. In this case the stochastic state space equations for the linear system are (Franklin et al.1990, p.444-445)

$$\begin{aligned} x(k+1) &= A_d x(k) + B_d u(k) + w(k) \\ y(k) &= C_d x(k) + v(k) \end{aligned} \quad . \tag{4.19}$$

In state space model (4.19) *w(k)* is a process noise and *v(k)* is a measurement noise. As (Burl 1999 p.218-221) presents the optimal solution for the stochastic system is equal with LQR solution.

4.3.1 Selection of the weighting matrices

For the control result to be satisfactory, the weighting matrices *Q* and *R* need to be defined. Because there is no universal way to select these matrices, the matrices need to be defined by using the trial and error method. There are some guidelines, which can help to select these weighting matrices.

One-way to select the weighting matrices is use to Bryson rule (Franklin et al. 1990, p.459). In this rule, the weighting matrix Q is selected as

$$Q = C_d^T Q_P C_d$$

$$Q_P = \begin{bmatrix} \frac{1}{m_1^2} & 0 & \dots & 0 \\ 0 & \frac{1}{m_2^2} & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \dots & & \frac{1}{m_n^2} \end{bmatrix} \tag{4.20}$$

where $m_1 \dots m_n$ are the maximum tolerances of the outputs. Equation shows that this rule focuses only for the weights of the outputs. Respectively, the weighting matrix for the control signals is

$$R = \begin{bmatrix} \frac{1}{u_{1m}^2} & 0 & \dots & 0 \\ 0 & \frac{1}{u_{2m}^2} & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \dots & & \frac{1}{u_{nm}^2} \end{bmatrix} \tag{4.21}$$

where $u_{1m} \dots u_{nm}$ are maximum values for the control signals. Now the cost function of the system is

$$J = \sum_{k=0}^{\infty} \rho x(k)^T Q x(k) + u(k)^T R u(k) \tag{4.22}$$

where ρ is a gain which affects the ratio between the output and state variables. This gain needs to defined by the trial and error method. As Equations (4.20 - 4.22) show, if maximum values in weighting matrices are selected right, there is only one gain to be determined.

4.4 Kalman filter

Like the tuning of the state feedback controller for the MIMO system, also tuning the estimator could be quite difficult. For this reason, the optimal estimator tuning method, which is known as Kalman filter is usually used. This filter provides optimal estimation for LQG problem (Grewal 1993 p.1) by giving minimizing the mean square estimation error (Burl 1999 p.231). In this problem it is assumed that noise, which affects the system, is Gaussian white noise.

It is possible to define the tuning gain matrix, which depends on time, for the estimator. Usually the constant gain matrix is sufficient, because the system runs a long time and past transients have no effect on next transients. The steady-state gain matrix for the estimator can be determined by observing the stochastic system, which presented in Equation (4.19). In this case both of the system noises can be assumed to zero meaning. The tuning equations expect that covariance of these noise are known. These covariance matrices can be presented by using the matrices *W* and *V*.

Estimator gain matrix can be calculated by using equation (Grewal 1993 p.112)

$$L = \bar{X} C_d^T (V + C_d \bar{X} C_d^T)^{-1} \tag{4.23}$$

where $\bar{X}$ is the state variance matrix. This matrix can be defined by using discrete time algebraic Riccatin equation (Franklin et al.1990, p.431)

$$\bar{X} = A \bar{X} A^T - A \bar{X} C^T (C \bar{X} C^T + V)^{-1} C \bar{X} A^T + W . \tag{4.24}$$

In Matlab, Equations (4.23) and (4.24) can be calculated by using *Kalman*-function, which calculates the gain matrix for the state estimator.

4.5 Reference inputs for the state feedback controller with estimator

The state feedback controller drives all system states to zero as a default, but usually the steady state value of all variables is not zero. That is why the reference input should be given to the state vector. The most common structure (Franklin et al. 1990, p.278) to define the reference values for the state feedback controller is presented in Figure 4.4.

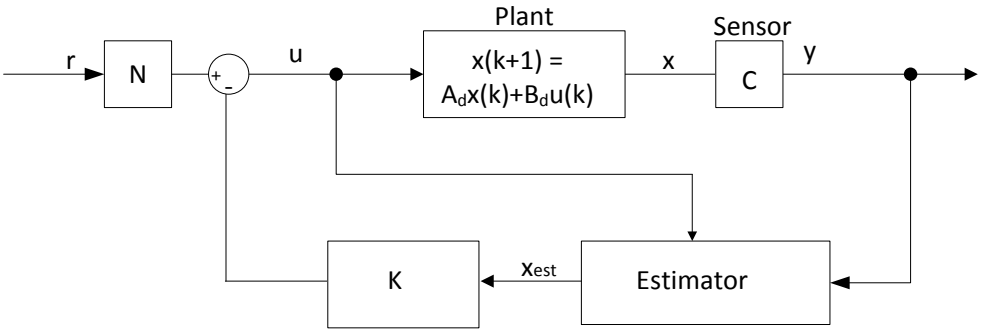

Figure 4.4 *Reference input for the state feedback controller with the estimator*

In Figure 4.4 N is a matrix which is defined (Franklin et al. 1990, p.277) as

$$N = N_u + KN_x \tag{4.25}$$

where N_x is matrix which is defined so that states in equilibrium point correspond to reference input r. N_u is a proportional matrix which is determined so that steady state control correspond to the reference input.

Matrices N_x and N_u can be defined by using equation

$$\begin{bmatrix} N_x \\ N_u \end{bmatrix} = \begin{bmatrix} A - I & B \\ C_r & 0 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{4.26}$$

where C_r is the output matrix which includes the states to be controlled. This equation is a suitable for Single-Input-Single-Output (SISO) and MIMO systems. This equation assumes that the number of the inputs and the number of the desired outputs is the same because only then there is a unique solution for this equation. If the system has a zero at the point *1 + 0i* the equation cannot be solved.

4.6 Structure of the anti-sway control system

This section presents the structure of the anti-sway controller, which was designed by using the provided theoretical background. Figure 4.5 presents the architecture of the anti-sway controller (Honkakorpi 2013, p.5-7).

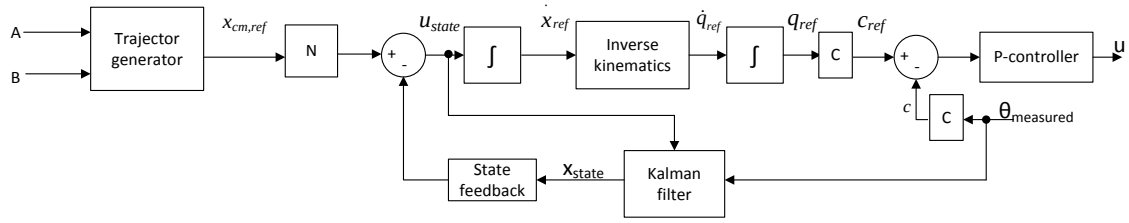

Figure 4.5 *The architecture of the anti-sway controller*

Figure 4.5 depicts that the anti-sway controller consists of the state feedback controller, Kalman filter and proportional controllers (p-controllers). The working principle of this control system is that the state feedback controller calculates the reference velocity for the boom tip. This calculation needs state variables, which Kalman filter estimates from the joint angles and positions and the reference signals, which are obtained from the trajectory generator. By using inverse kinematic functions, the reference velocity of the boom tip transforms to the reference velocities of the joints. By integrating the reference velocities of joints the reference positions for the joints are obtained. These reference positions can be transformed to reference positions the hydraulic actuators. According to this reference the control signals to the valves is calculated by p-controller.

In Figure 4.6 the structure of the anti-sway controller by using Matlab blocks is presented.

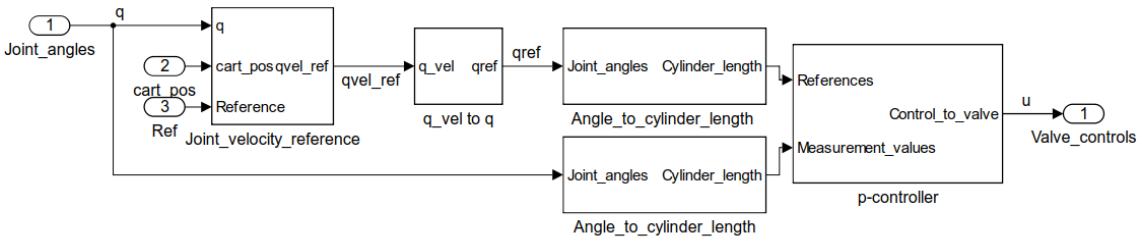

Figure 4.6 *The high-level architecture of the controller*

As Figure 4.6 depicts there are four different parts in the high-level. The low-level controller of this system is a basic p-controller (Dorf et al. 2005, p.391), which is presented in Figure 4.7.

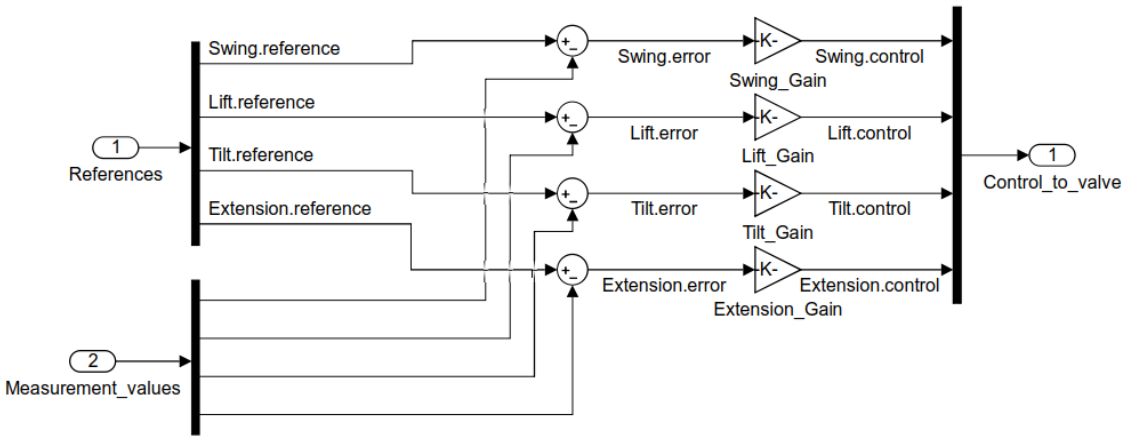

Figure 4.7 *The p-controller*

In hydraulics PI-controller is not used because the hydraulic cylinders can be modeled as an integrator. This controller calculates the error between the reference and the real value and products that with desired gain. The result of this product is the control signal to valves.

In Figure 4.7 cylinder lengths can be defined directly from the joint variables by using the matrix 2.11. The joint variables are obtained by integrating velocities which is presented, in Figure 4.8.

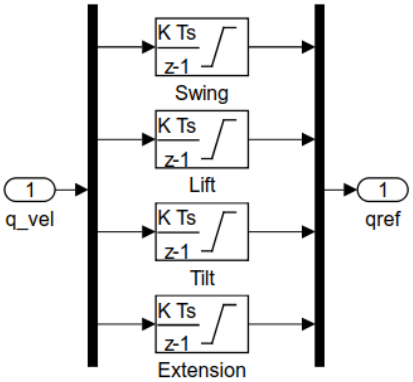

Figure 4.8 *Integrators with saturation*

Figure 4.8 shows that the values of joint variables need to saturated to correspond to the real system.

The reference velocities for the joint variables can be calculated by using the state feedback controller and the Jacobian inverse matrix. The architecture of the anti-sway controller in Figure 4.5 can be implemented by using basic Matlab blocks which is presented in Figure 4.9.

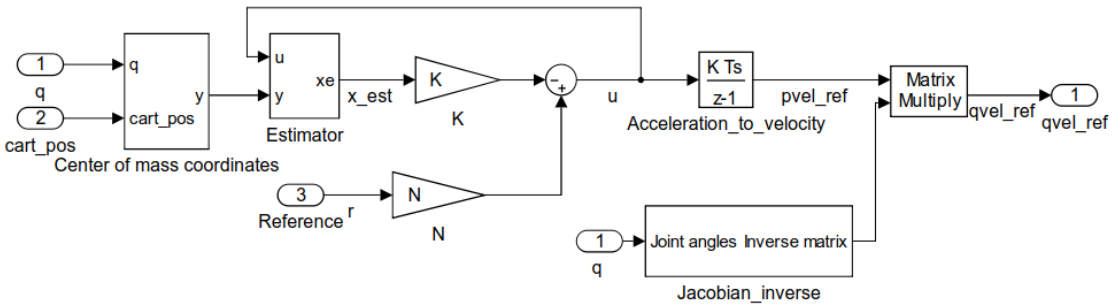

Figure 4.9 *The architecture of the anti-sway controller*

As Figure 4.9 shows the anti-sway controller consists of the estimator, the state-feedback controller and inverse matrix of Jacobian.

5. MEASUREMENTS

This chapter presents the measured results of the designed the anti-sway on to the crane. For comparison open-loop control results are also presented. At the beginning of this chapter the verification of the dynamic model and the verification of Kalman filter based estimation are presented. After these verifications the measurement results for the closed-loop and the open-loop control with masses 100 kg and 250 kg are presented.

A real time control interface to the sensors and servo valves is implemented by using PowerPC-based dSpace DS1103 system. The sampling rate of the controller is 500 Hz.

5.1 Dynamic model verification

The control system design requires the dynamic model of the tool as was developed in Chapters 3 and 4. Because the control system design is based on the dynamic model it is important that the model is as accurate as possible. For that reason, the dynamic equations (3.12 - 3.14) need to be tuned to correspond to the real system. As Chapter 3 presented, the tuning of these equations can be done by fixing the friction and the mass moment inertia parameters. The tuning of these parameters can be done by comparing the model with the real system.

The viscous friction parameters can be tuned by comparing the dynamic model with the real system. At first the situation when the tool swings separately in both directions is studied. When the acceleration of the boom tip is zero only the friction affects the damping and the parameters can be tuned. Because the mass moment inertia of the real tool is different in both directions, the mass moment inertia has to be fixed to compare to the real system. The mass moment inertia affects the swaying frequency of the tool and the friction parameters affect damping. Figure 5.1 presents the real tool measurements versus the dynamic model motion estimation, when the tool swings only in one direction at a time so in Figure 5.1 is presented two different tests.

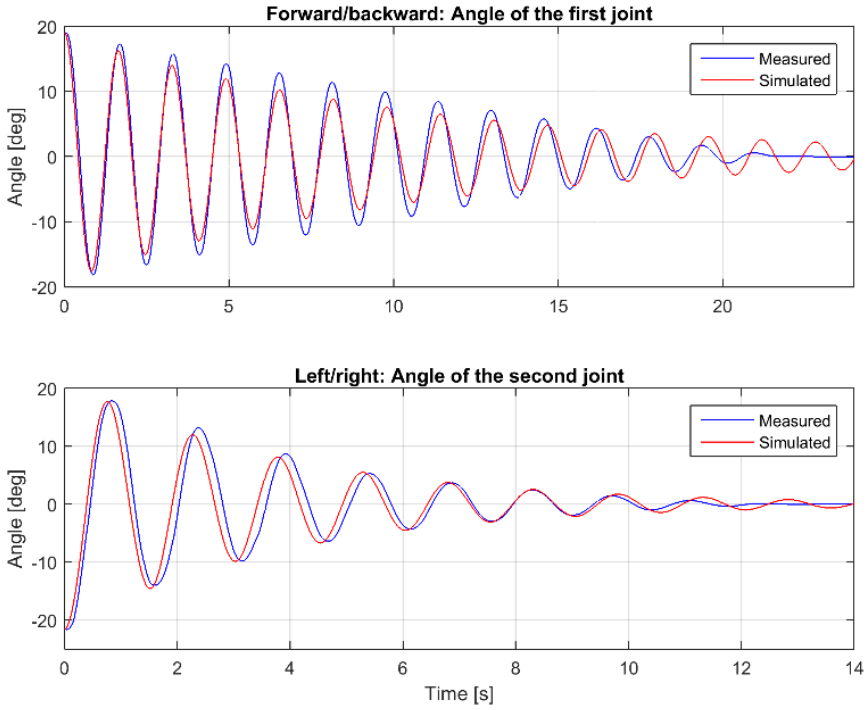

Figure 5.1 *The dynamic response of the tool*

The first subfigure in Figure 5.1 depicts the dynamic response of the tool, when the initial angle for the first joint is 20° and the tool is allowed to swing freely. The second subfigure shows the dynamic response, when the initial angle for the second joint is -20° and the tool is allowed to swing freely. The figure shows that the dynamic model corresponds quite well to the real system. The main difference is that the damping of the model is not as high in reality. When the tool swings left/right, the damping is almost the same as the real but when the tool swings forward/backward the damping is much lower. The damping of the model is lower, because higher friction parameters cause error at higher amplitudes. Because the friction model is linear there need to make compromise about the friction gain.

Figure 5.1 verifies only the equations of the angular accelerations. Another part of the dynamic model are the linear acceleration equations that also needed to be verified. Figure 5.2 presents the center of mass coordinates, when the tool swings only to forward/backward and left/right swaying is nearly zero.

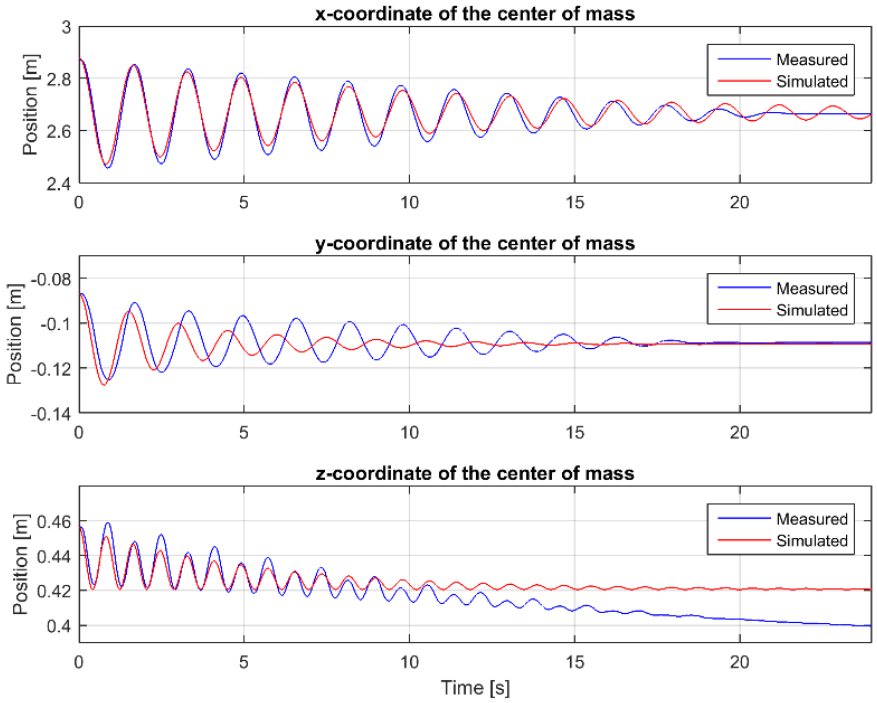

Figure 5.2 *The coordinates of the center of the mass*

Figure 5.2 depicts that *x*-coordinate model corresponds well to the real system. Also *z*-coordinate model corresponds to the real system. The drifting of *z*-coordinate is due to the leakages of the extension boom valves. Figure shows that *y*-coordinate model does not correspond very well to the measured values. At the beginning *y*-coordinate model corresponds quite well to the measured coordinate but then the error starts to rise. This error is due to the damping of the model. Therefore, it is possible to deduce that with the low amplitude the model does not correspond to the real tool but the error is not significant.

Respectively Figure 5.3 presents the coordinates when the tool swings only left/right and forward/backward movements are low.

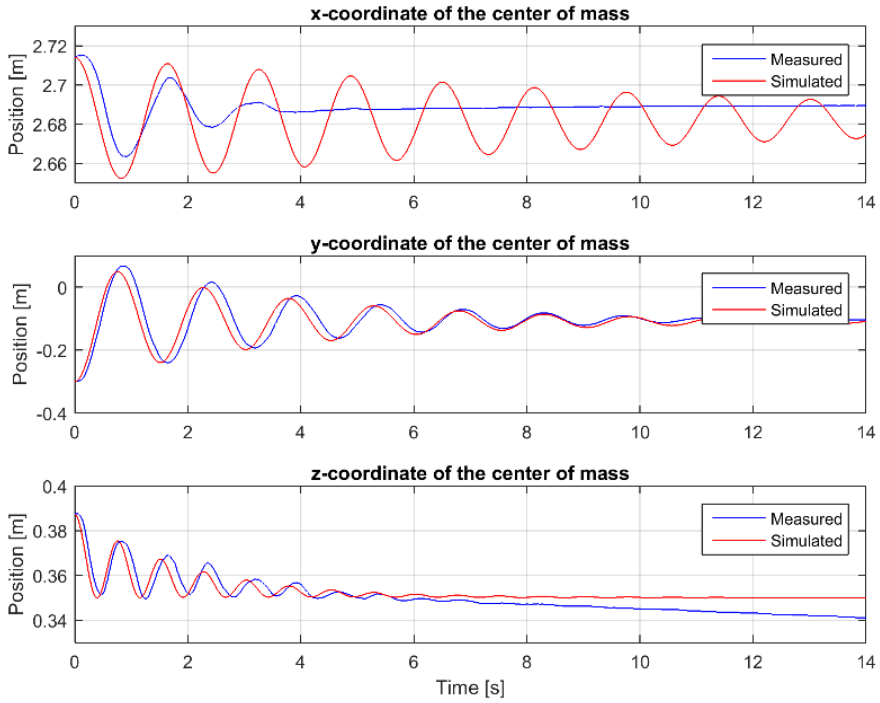

Figure 5.3 *The center of mass coordinates*

Figure 5.3 depicts that *y*- and *z*-coordinate models correspond quite accurately to the real system. The Figure shows that *x*-coordinate model does not correspond very well with measured value. The *x*-coordinate model does not correspond to the real system because the damping of the model is smaller than the real damping. For that friction corresponds to real system need friction model which depends on amplitude. Because LQR needs linear model, the friction model should be linear.

Another way to verify the dynamic equations is to compare the model with the real system when the tool swings simultaneously in both directions. Figure 5.4 presents the dynamic response of the tool in this case.

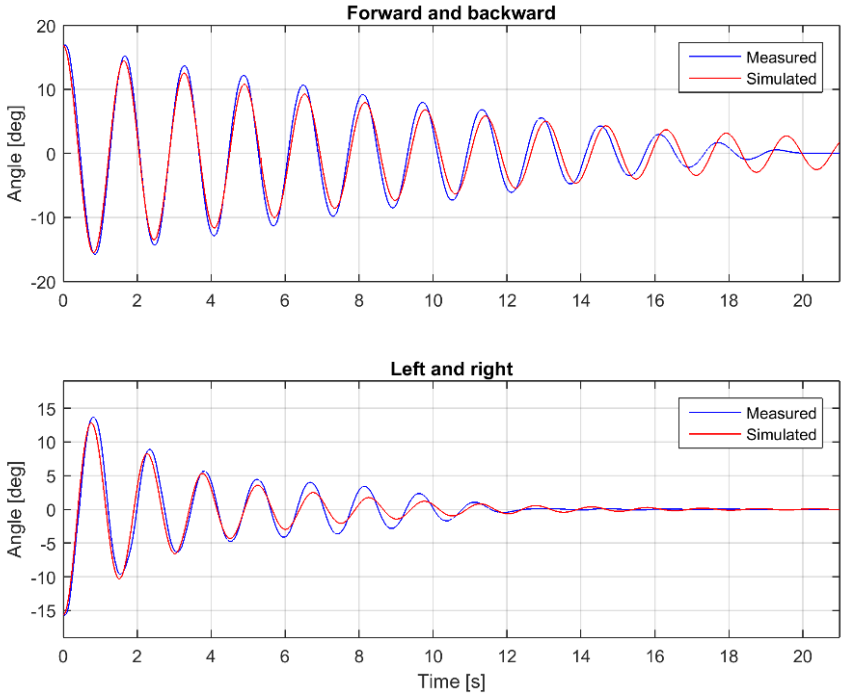

Figure 5.4 *Angles to both directions*

Figure 5.4 shows that the friction parameters and the mass moment of inertia parameters correspond to the real system. Respectively the coordinates of the center of mass are presented in Figure 5.5.

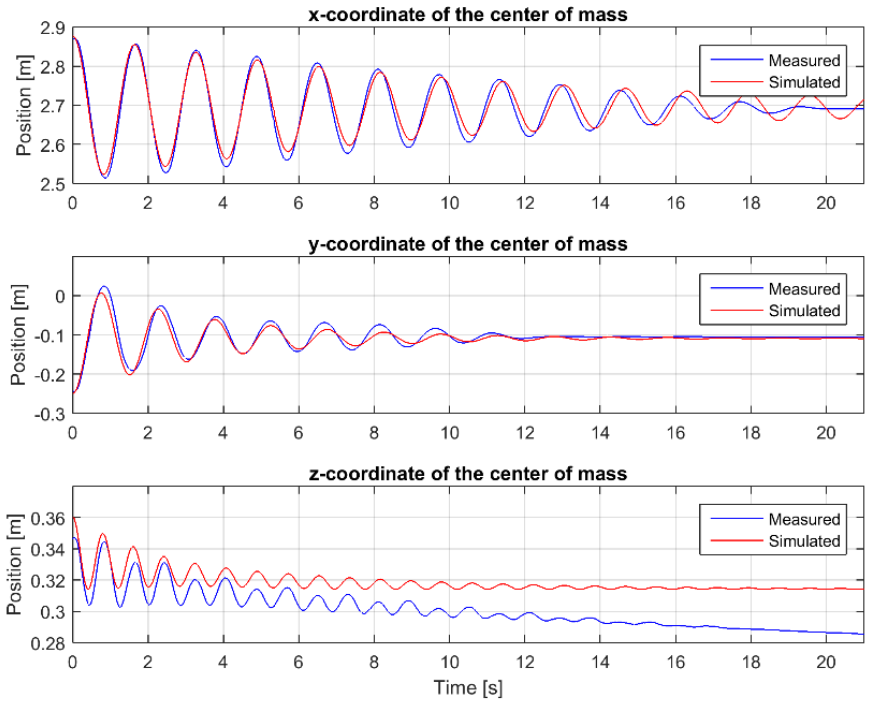

Figure 5.5 *The center of mass coordinates*

Figure 5.5 depicts that also the real and the simulated coordinates correspond well to each other.

According to Figures 5.1 – 5.5 it can be deduced that the dynamic model corresponds well to the real tool. Hence it is possible to design a controller by using these equations. Only modeling errors are caused by small damping of the dynamics model.

5.2 Kalman filter verification

Chapter 4 presented that Kalman filter design includes the dynamic model of the tool. As presented in Equation (4.23) a Kalman filter needs to be tuned to correspond the real system. This tuning can be done by fixing the covariance matrices of the system so that the estimator produces accurate estimates for the state variables.

Because the data from the incremental encoders does not include noise, the values for the covariance matrices are small. In this thesis the diagonal values of the covariance matrix W in Equation (4.23) are 0.001 and the diagonal values of the covariance matrix V in Equation (4.24) are 0.01. These covariance matrices were selected according to experimental tests. The tuning functions for Kalman filter are presented in Appendix G. The estimator accuracy can be verified by comparing the input signals with the filtered output signals. Figure 5.6 presents these signals when the tool swings simultaneously in both directions.

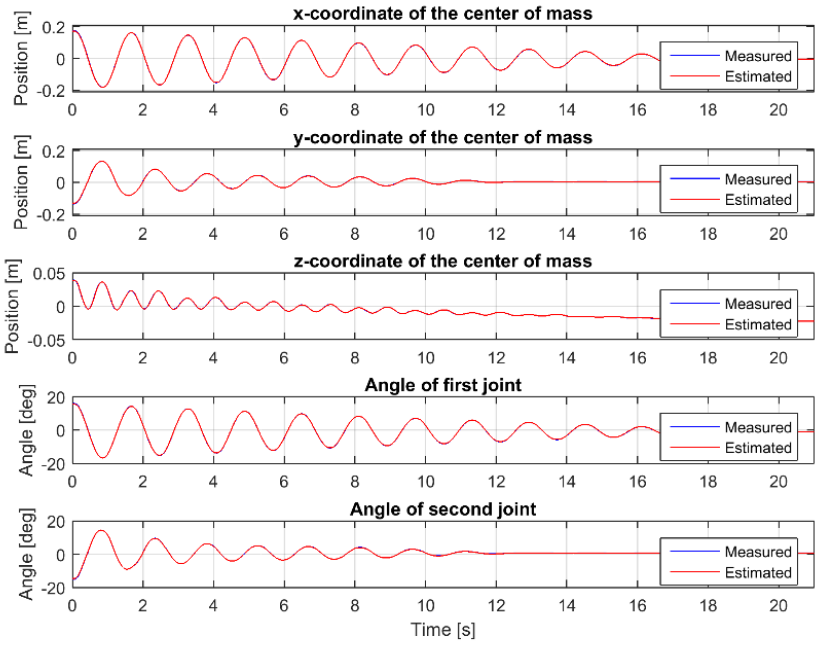

Figure 5.6 *Measured signals and filtered signals*

Figure 5.6 shows that the measured signals correspond well to filtered signals. Because the measured signals do not include noise a filter does not affect these signals much. Because Kalman filter is based on the linearized model the filter estimates only the difference from the equilibrium point and not directly from the center of mass position. The joint angles at the equilibrium point are zero so the estimated angles are the joint angles.

In figure 5.7 the differences between the input and output signals are presented.

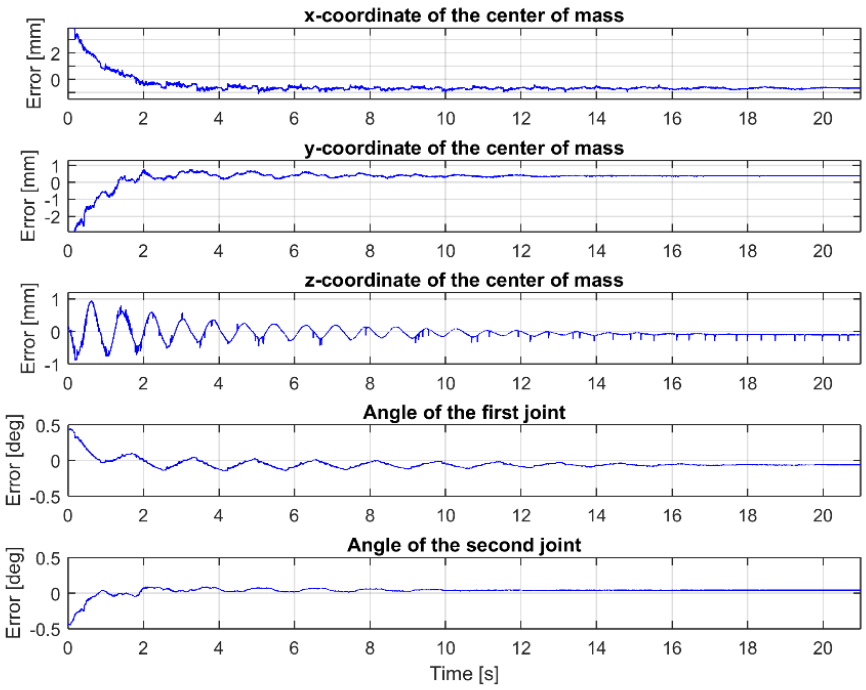

Figure 5.7 *Difference between input and output signals*

Figure 5.7 depicts that differences between input signals and filtered output signals are very low. As the Figure depicts the difference is higher at the beginning than at the end because the initial state of the estimator is not the same as the initial state of the real tool. Also the modeling error of the linear system affects the difference. The noise of these differential signals describes the noise which the filter removes.

Because the measured signals do not include much noise, the main purpose of Kalman filter in this system is to estimate the missing state velocities. It is also possible to define these velocities by differentiating position values of state variables, but usually this produces noise to the variables. If the position signals already include noise, the noise level in velocity signals rise and these signals require a filter. Kalman filter produces already optimal estimation for the velocities. Figure 5.8 presents the estimated velocities of the position signals.

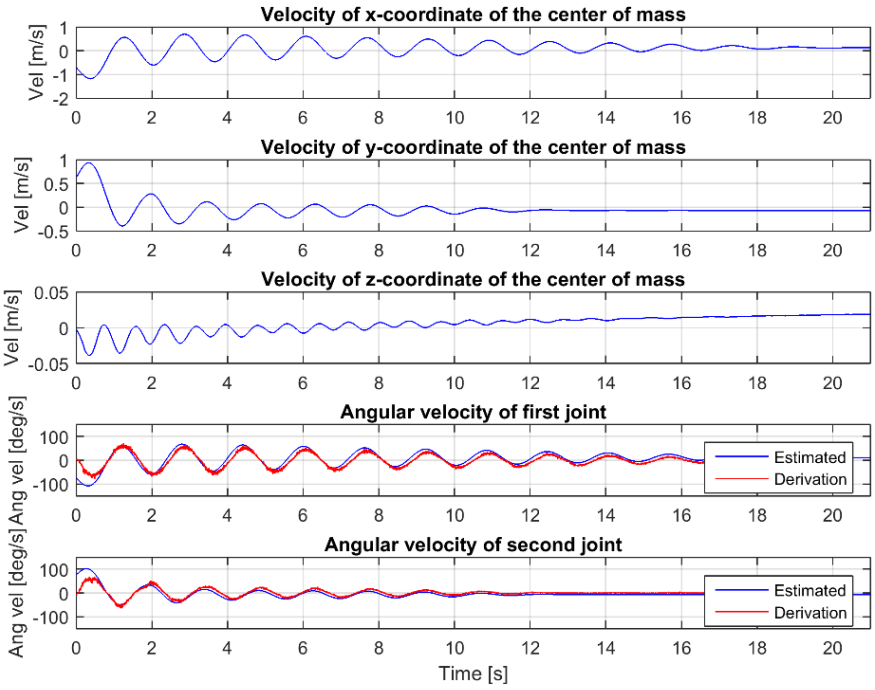

Figure 5.8 *Estimated state velocities*

Figure 5.8 depicts that estimator can produce the values of the missing state variables. In addition, the Figure shows that the angular velocities of the joints correspond quite well to the real system. There is a small difference between the differentiated and the estimated signal, because the linear estimator cannot handle all nonlinear frictions of the system.

According to Figures 5.6 and 5.8 it is possible to say that Kalman filter can estimate the missing state variables and filter the measurement signals. Because Kalman filter could produce the missing state values state feedback controller can be use in this system.

5.3 The anti-sway controller for the tool

This section compares the system with the anti-sway controller and without controller, when mass of the tool is just the mass of the gripper. So in this case the mass of the tool is about 100 kg. The hydraulic power supply is set in 20 MPa supply pressure and the maximum flow from the pump is 90 l/min.

The reference paths to the controller are generated by using a five-degree polynomial, which is presented in Appendix H. In this thesis two different kind of test path for the controller are used. The first path is a *xz*-path which gives reference just to the *x*- and *z*-coordinates and the reference to *y*-coordinate are constant. The second path is a *xy*-path

where the reference to *z*-coordinate is constant and the reference to *y*-coordinate change. Both of these paths can be given with a different duration. Figure 5.9 presents positions of these paths in a workspace of Hiab 033.

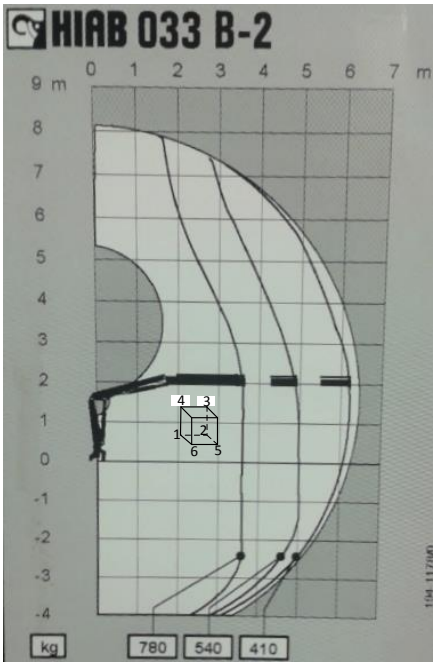

Figure 5.9 *The reference paths in the workspace*

The *xz*-path goes through points 1-2-3-4 in the Figure and the *xy*-path goes through points 1-2-5-6. Figure 5.10 presents reference points for the center of mass coordinates.

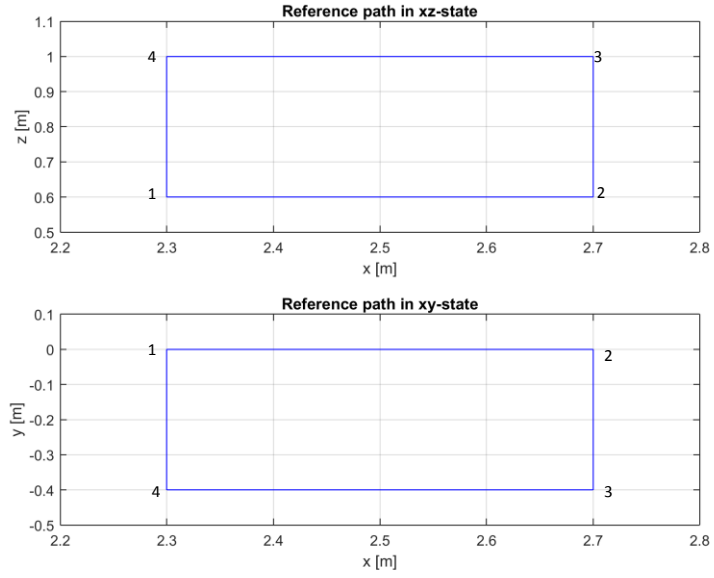

Figure 5.10 *Reference paths*

Figure 5.10 shows that both of these reference paths are square.

Chapter 4 presented the LQR tuning method for the state feedback controller. In this case the controller was tuned so that an overshoot and a settling time of the tool are minimized. The tuning matrix, which produces the best results, is obtained by giving more weight for the outputs than for the control signals. The tuning matrix, which was used in this thesis, is

$$K = \begin{bmatrix} 8.4362 & 5.9727 & 0 & 0 & 0 & 0 & -10.9311 & -3.7605 & 0 & 0 \\ 0 & 0 & 4.2258 & 3.4971 & 0 & 0 & 0 & 0 & -4.3855 & -2.0849 \\ 0 & 0 & 0 & 0 & 9.3872 & 4.333 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.1}$$

If the values in the tuning matrix are any higher the controller cannot handle all nonlinear frictions and the system start to oscillate. The tuning of the state feedback controller can be done by using the functions shown in Appendix G.

The first case compares the controlled and the uncontrolled system when the extension boom is unused. Without the extension boom, all the movements of the crane are done by using the lift and tilt boom. In this case the reference path used is the *xz*-path. Figure 5.11 presents the center of mass coordinates and joint angles when the duration of the path is 7.2 second and velocity of the boom tip is 0.22 m/s.

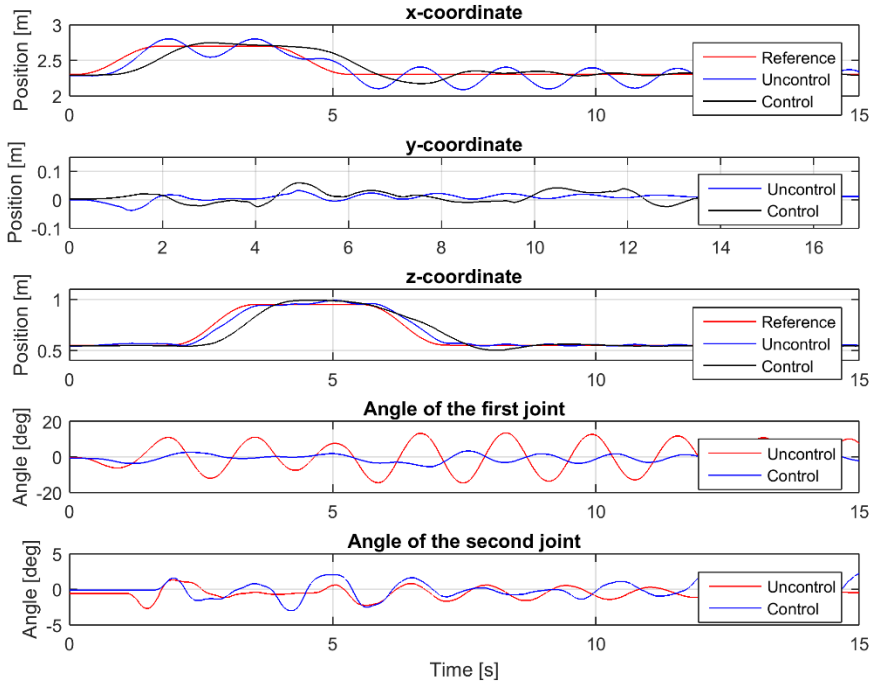

Figure 5.11 *The center of mass coordinates and the joint angles*

Figure 5.11 depicts that the anti-sway controller can compensate the swaying of the tool quite efficiently and the maximum overshoot is only a few percent. Also the settling time of the controlled system is about 2 seconds. Figure shows that the tracking error of the anti-sway controller is larger than the error of the uncontrolled system. Because the

controller does not react for low state errors, there is some delay in the control system. In theory it is possible to tune the anti-sway controller to react faster to reference, but in practice this cause, instability to the system. Because in the crane there are nonlinear frictions, which the controller cannot handle, the system can become unstable.

In figure 5.12 angles and velocities of the swing, the lift and the tilt joints when the anti-sway controller is connected are presented.

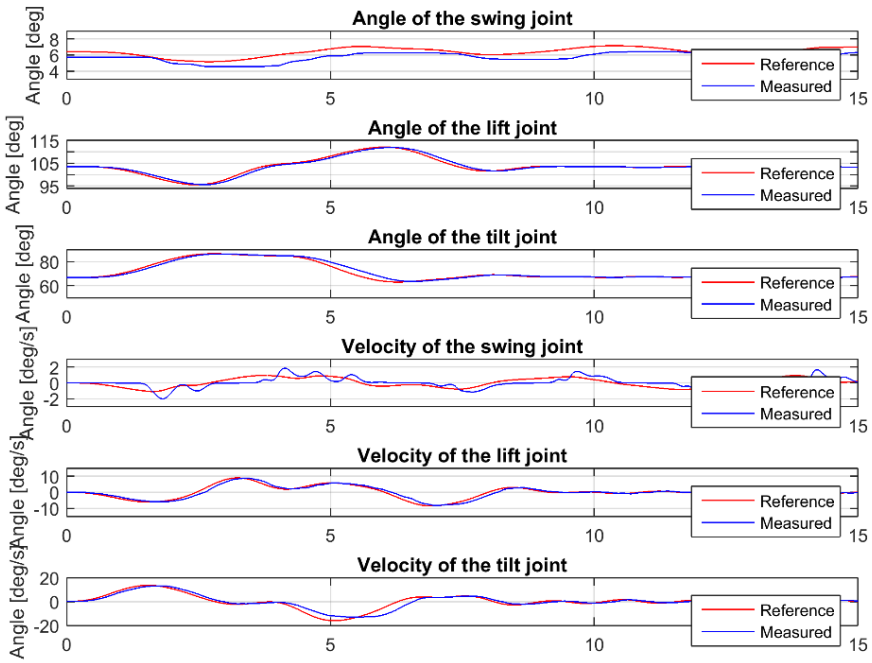

Figure 5.12 *Joint angles and velocities*

According to joint angles the reference for the p-controllers is controlled. Figure 5.12 shows that the tilt joint and lift joint follow the reference quite accurately. The maximum error in lift and tilt joints is about 3°. The swing joint follows reference less accurately, but the maximum error is about 3°. Because the boom tip is far from the base coordinate frame, a small error in swing joint cause significant error to Cartesian position.

Figure 5.13 presents the control signals to the lift and tilt valves.

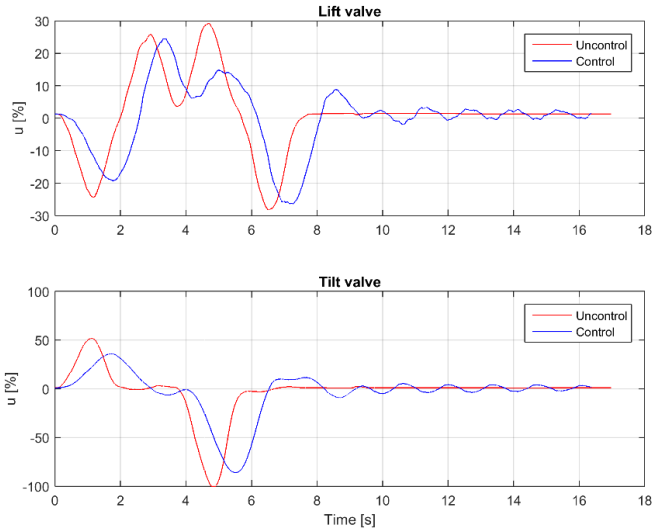

Figure 5.13 *The control signals to the lift and tilt valves*

By comparing Figure 5.13 with Figure 5.11 it can be seen that the control signal for the tilt valve is almost the maximum when the cylinder moves backwards. For this reason, it is not possible to give any significant faster reference paths.

Because using just the tilt and lift booms cannot give any faster reference paths the extension boom is needed. Figure 5.14 presents the same test as Figure 5.11 but with the extension boom.

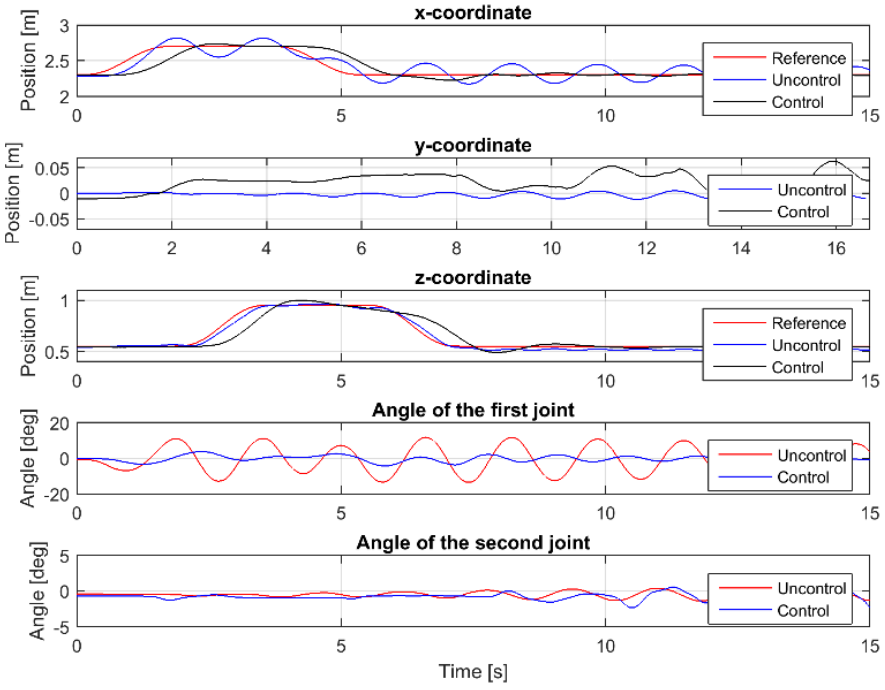

Figure 5.14 *The center of mass coordinates and the joint angles*

Figure 5.14 shows that the swaying of the tool can be compensated with the extension boom and the overshoot and the settling time do not change. In Figures 5.11 and 5.14 angle of the first joint sways a lot because the acceleration of boom tip affects the tool a lot.

When the extension boom is used, control signals to lift and tilt valves are smaller than without the extension boom. Figure 5.12 presents control signals to all valves.

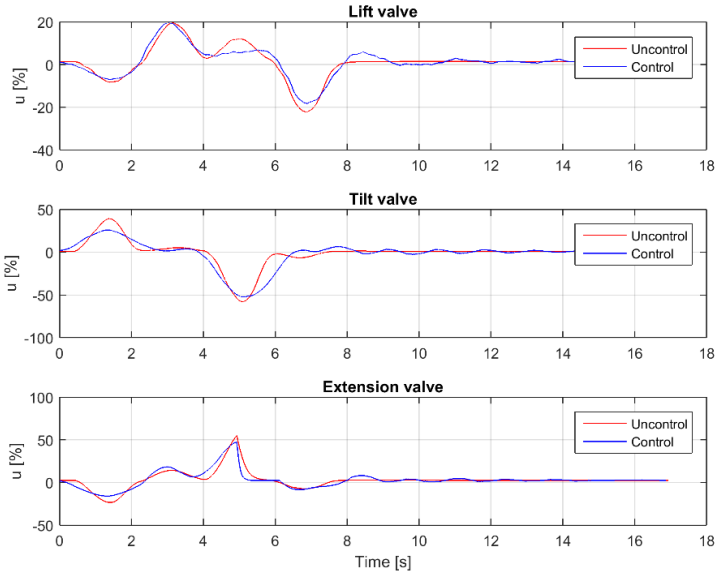

Figure 5.15 *Control signals to the lift, tilt and extension valves*

As Figure 5.15 shows control signals are lower than in Figure 5.13 and any of these signals are not near to full opening. Therefore, it is possible to give faster reference path for the crane.

Next figures present the quality of the anti-sway controller with some faster reference paths. As Figures 5.11 and 5.14 show the open-loop controlled tool sways already with the slower reference so for clarity only the controlled system was presented. Figure 5.16 presents the system behavior when the duration of the path is 5.2 seconds and velocity of boom tip is 0.3 m/s.

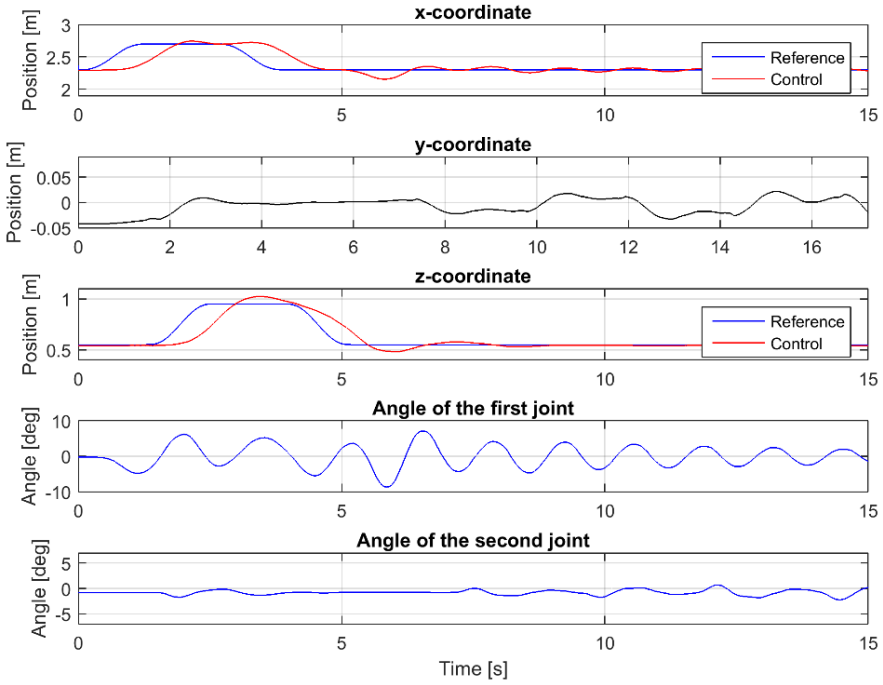

Figure 5.16 *The center of mass coordinates and joint angles*

Figure 5.16 depicts that the controller can compensate the swaying quite well and the overshoot of the tool is only a few percent. As the Figure shows the controller cannot drive the joint angles to absolute zero but the controller can hold these angles low. As the Figure shows the anti-sway controller cannot hold *y*-coordinate absolutely constant but error is only a few centimeters. The error in *y*-coordinate is caused by nonlinear flexibility of the swing joint.

Figure 5.17 presents the quality of the anti-sway controller when the duration of the path is 4 seconds and velocity of boom tip is 0.4 m/s.

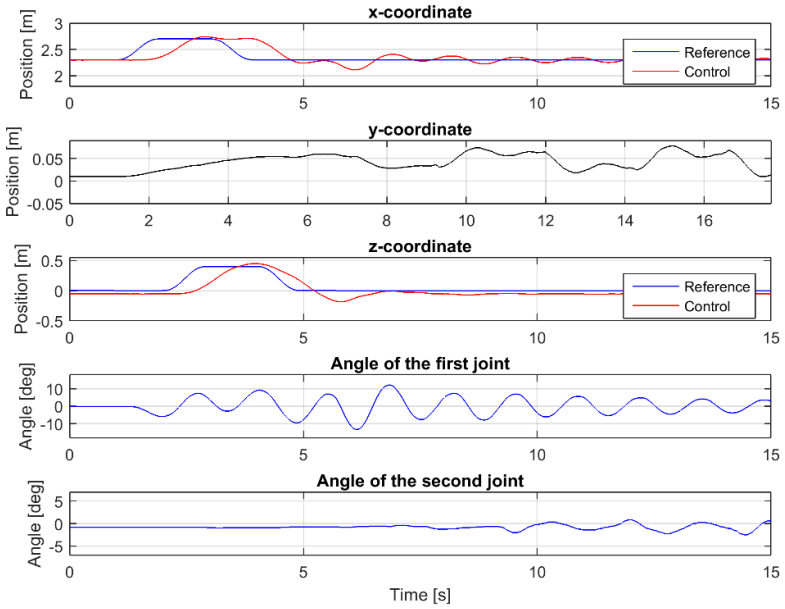

Figure 5.17 *The center of mass coordinates and joint angles*

As shown in Figure 5.17 the controller can stabilize the tool also with this path. The overshoot grows little and the controller cannot remove all swaying of the first joint as fast as with slower reference.

Previous Figures presented the quality of the anti-sway controller with *xz*-path. Next Figures present the anti-sway controller quality with *xy*-path. Like in Figure 5.13 also with *xy*-path control signal to the tilt valve will be saturated so the extension boom is used. Figure 5.15 presents the quality of the anti-sway controller when the duration of path is 7.2 seconds and velocity of boom tip is 0.22 m/s.

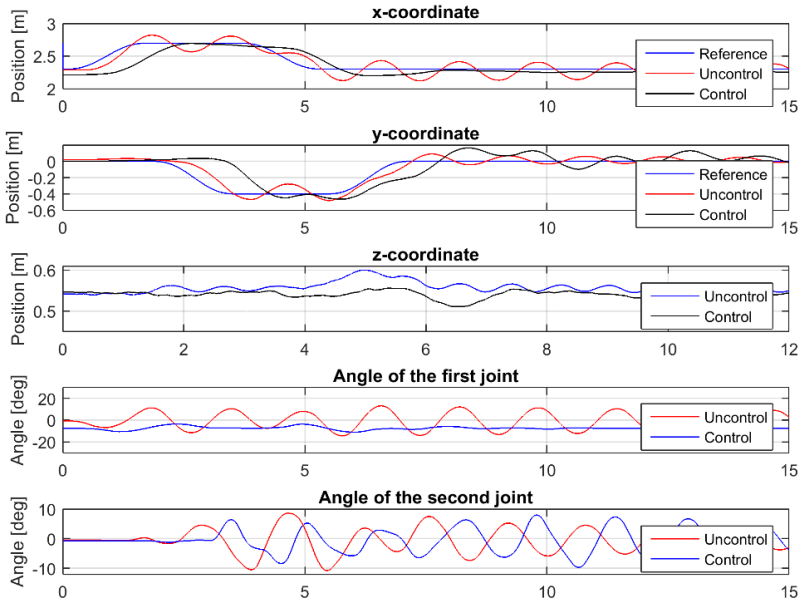

Figure 5.18 *The anti-sway controller with xy-path reference*

As Figure 5.18 shows, the anti-sway controller cannot compensate all the swaying of *y*-coordinate because the p-controller cannot handle the nonlinear flexibility of the swing joint. The controller can compensate the swaying of the *x*-coordinate and keeps *z*-coordinate almost constant.

Figure 5.19 presents the quality of the controller when the duration of the path is 5.6 second and velocity of boom tip is 0.28 m/s.

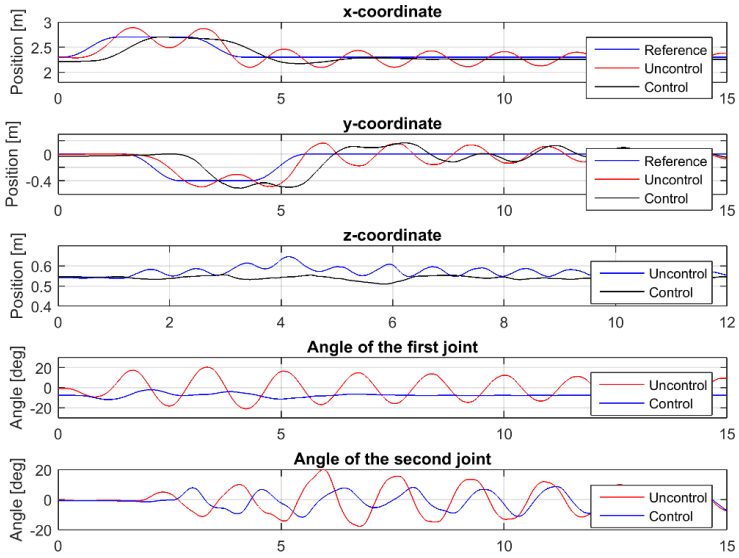

Figure 5.19 *The center of mass coordinates and joint angles*

As Figure 5.19 shows, the anti-sway controller can keep the swaying of the second joint smaller than without a controller, but still *y*-coordinate sways quite much and the tracking error is quite large.

In summary it can be said that the anti-sway controller can compensate for the swaying of the *x*-coordinate efficiently and when the settling time is short. With *xz*-path the controller can compensate the swaying of the *z*-coordinate. The controller can also quite well keep y-coordinate constant with *xz*-path. With *xy*-path the controller cannot compensate all the swaying of the *y*-coordinate but other coordinates are stable.

5.4 Anti-sway controller for the tool with load

The previous section presented the behavior of the anti-sway controller when the mass of the tool was just the mass of the gripper. Because for most of the time the tool moves heavy loads the controller needs to be tested with a load mass. In this thesis, a load mass is about 150 kg so the total mass of the tool is now 250 kg.

Figure 5.20 presents how the load mass affects the dynamic response of the tool.

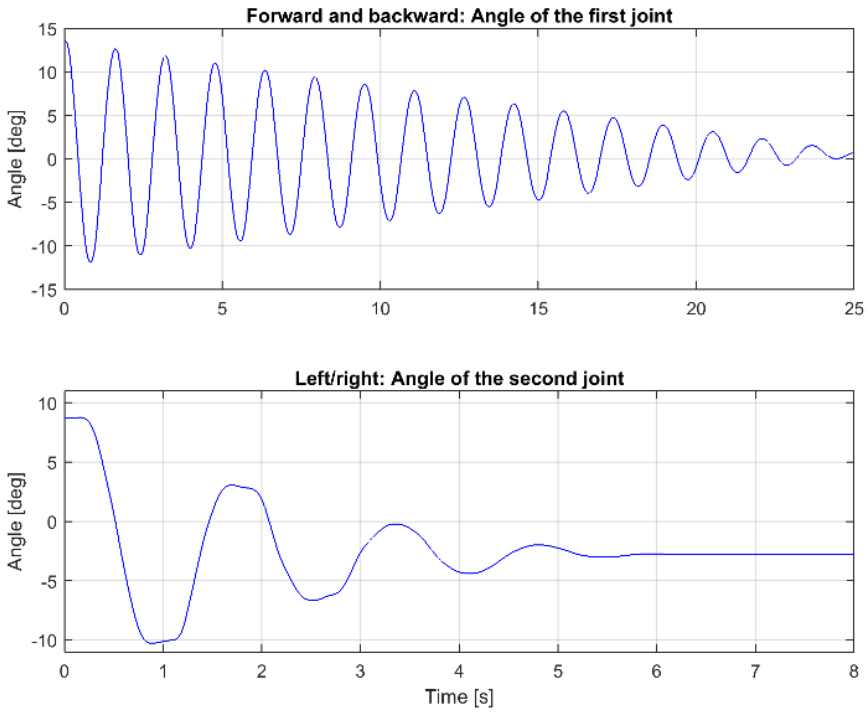

Figure 5.20 *Dynamic response of the tool with the load mass*

Figure 5.20 shows that the load mass does not affect forward/backwards movements. By comparing the natural frequency of the tool in Figure 5.1 with Figure 5.20 it can be

noticed that the frequencies are almost the same. The natural frequency of this tool is about 0.63 Hz. As the Figure shows the tool is not as ideal as without the load when the tool sways left/right. Because the load mass does not affect the natural frequency the same controller than above can be used.

Like presented in the previous section without the extension boom the tilt valve saturates easily so in this section only the system with the extension boom is studied. Figure 5.21 presents the controlled tool with the same reference path as in Figure 5.11. The velocity of boom tip is 0.22 m/s.

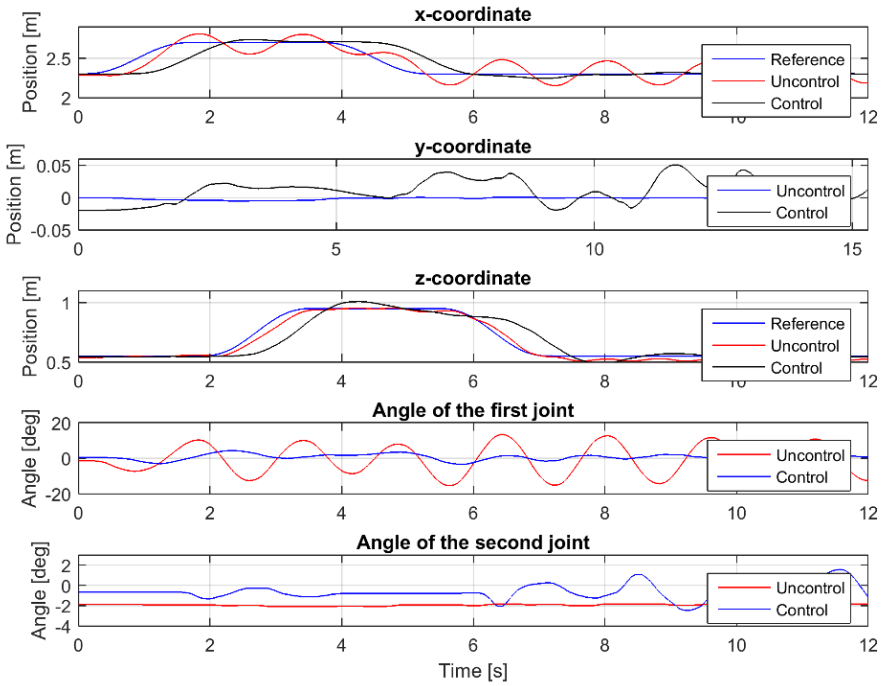

Figure 5.21 *The controlled tool with higher load mass*

Figure 5.21 shows that the anti-sway controller can compensate the swaying of the tool with the load mass. The overshoot and settling time are almost the same than without the load. By comparing Figures 5.21 and 5.14 it can be defined that the load affects only the swaying of *y*-coordinate but the coordinate is still near to constant.

Figure 5.22 presents the system when the velocity of boom tip is 0.3 m/s.

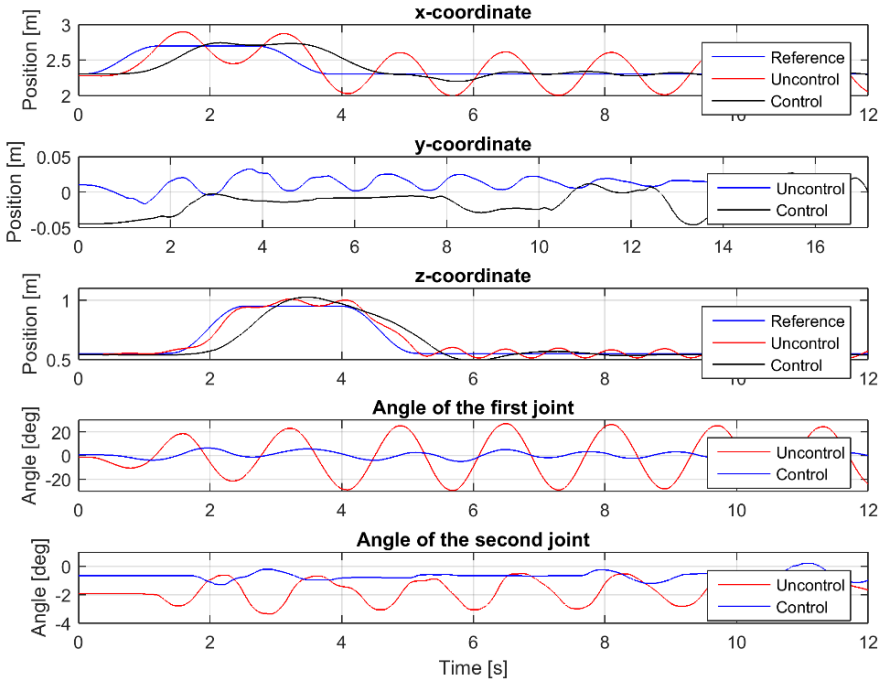

Figure 5.22 *Anti-sway controller with faster reference path*

Figure 5.22 shows that the controller can compensate the swaying of the system when the reference path is faster. By comparing Figures 5.22 and 5.13 it can be seen that in both cases the overshoot and settling time are almost the same.

Figures 5.21 and 5.22 present the anti-sway controller with *xy*-path and load. In Figure 5.23 the coordinates and angles when velocity of boom tip is 0.22 m/s are presented.

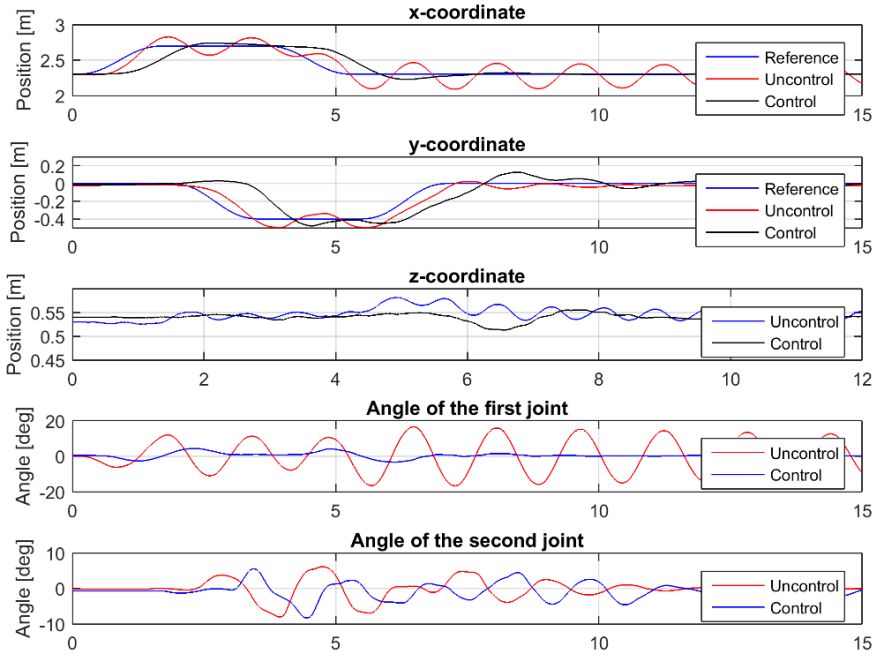

Figure 5.23 *The anti-sway controller with xy-path reference*

As Figure 5.23 depicts, the anti-sway controller can compensate the swaying of the tool quiet efficiently. By comparing Figure 5.23 to Figure 5.14 it can be seen that the load mass affects only the *y*-coordinate. As Figure 5.20 presented, the damping of the left/right movements is a little higher than the damping of without load. That is why *y*-coordinate does not sway as much as without a load mass.

Figure 5.24 presents the coordinates when the velocity of the boom tip is 0.3 m/s.

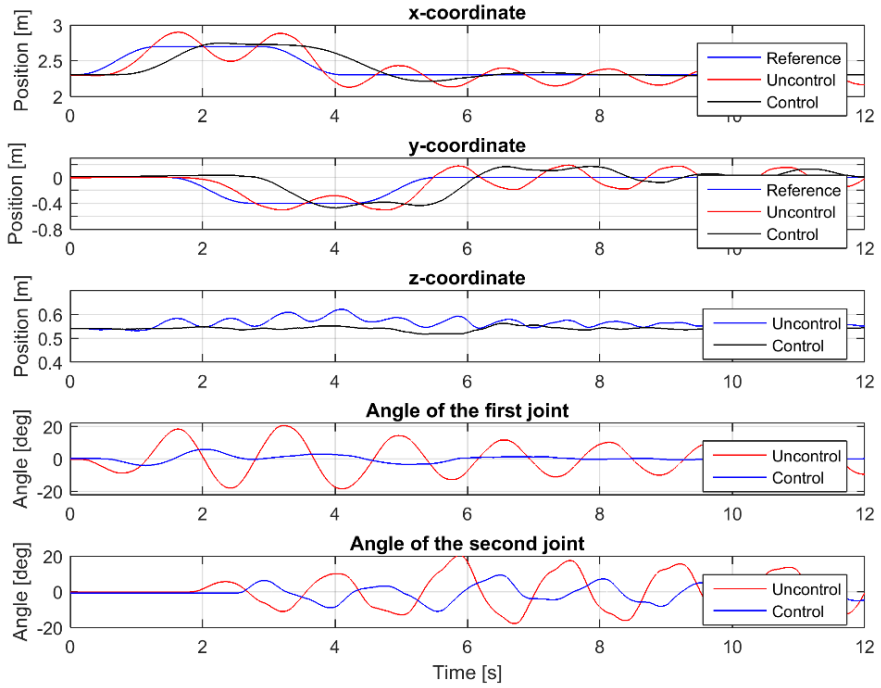

Figure 5.24 *The coordinates and joint angles*

Figure 5.24 depicts that also with a faster reference path the higher load mass affects only the swaying of the *y*-coordinate. The overshoot and settling time of the tool are almost the same than in Figure 5.16.

In summary it is possible to say that the anti-sway controller can compensate the swaying of the tool efficiently and the load mass does not affect to the controller. The result shows that the linear controller cannot handle all nonlinear frictions of the system so the use of some nonlinear controller to compensate all swaying of the tool might be needed.

6. CONCLUSIONS

The goal of this thesis was to design an anti-sway load control system for a hydraulic crane. In this thesis the swaying tool, which was connected to the crane, was a common used gripper. This gripper was connected to the crane by using two joints, which could sway freely in two different directions (2DOF). In this thesis the test crane was Hiab 033.

At the beginning of this thesis the forward kinematics and the differential kinematics equations for the target system were determined. These equations enabled the control the Cartesian position of the boom tip instead of the traditional joint control. The controller design also required the dynamic model of the tool. In this thesis, the dynamic model of the tool was determined by using a Lagrangian dynamic formulation. By comparing the model with the real system, the model was tuned to correspond to the real system. According to measurement results, the dynamic model corresponded well to the real system.

The linear control methods architecture for the anti-sway load control system was used, which consisted of two main parts. The first part was the state feedback controller, which gives the velocity reference for the Cartesian position of the boom tip. Because only the tool swaying angles were measured, the control system also included Kalman filter. This filter produces values for the missing velocity measurements. By using inverse kinematic the Cartesian velocity reference was transformed to reference velocities of the joints. The second part of the control system were proportional controllers that calculate control signals for the valves. The position references were determined by using transformation between the joint space and actuator space.

In this thesis the anti-sway controller was tested by using two different sizes of load masses and two different test paths. According to these measurements it can be said that the controller could compensate the tool swaying. When the tool swayed forward/backward, the controller compensated the swaying efficiently and without large overshoot. When the tool swayed left/right, the controller could not compensate all swaying of the tool. The reason for this is the nonlinear flexibility of the swing joint. According to measurement, it can be noticed that increasing the load mass does not affect the control results.

In future it is possible to use nonlinear low-level controller instead of p-controllers. The nonlinear controller in handling nonlinear flexibility of the swing joint. Because nonlin-

ear controller handles nonlinear frictions of the system better it is possible to give a faster reference path for the center of mass.

In summary it can be said that the linear anti-sway controller can compensate the swaying of the real 2DOF tool and the control system can be implemented by using just the swaying angle measurements.

REFERENCES

Ahmad, M., Ismail, R. & Ramli, M. (2009). Input Shaping Techniques for Anti-sway Control of a 3-D Gantry Crane System. 2009 International Conference on Mechatronics and Automation. pp. 2876 – 2881

Bajd, T., Mihelj, M., Lenarcic, J., Stanovnik, A. Munih, M. Robotics. (2010). Springer Science+Business Media B.V. pp. 151

Beiner, L., Mattila, J. (1999). An improved pseudoinverse solution for redundant hydraulic manipulators. Robotica 17, pp. 173-177

Burl, J.B. (1999). Linear optimal control: H2 and H∞ methods. Menlo Park (CA), Addison-Wesley. pp. 462

Dorf, R.C. & Bishop, R.H. (2005). Modern control systems. 10th ed. ed. Upper Saddle River (NJ), Pearson Prentice Hall. pp. 881.

Fraba. (2014a). Position and motion sensors. referred 4.8.2016. available: https://www.posital.com/media/posital_media/documents/catalog_2014/Posital_Sensor_Catalog_2014_en.pdf

Fraba. (2014b). Ixarc incremental encoder datasheet. referred 4.8.2016, available: https://www.posital.com/en/products/incremental-encoders/incremental-encoder-finder/UCD-INS00-XXXXX-D10D-ARW/112723201/detail.php#

Franklin, G.F., Powell, J.D. & Workman, M.L. (1990). Digital control of dynamic systems. 2nd ed. Addison-Wesley Publishing Company. pp. 837

GE measurement. Unix 5000 Pressure sensing platform. (2014) referred (5.8.2016). available:
https://www.gemeasurement.com/sites/gemc.dev/files/unik_5000_datasheet_english.pdf

Grewal, M.S. & Andrews, A.P. (1993). Kalman filtering: theory and practice. Englewood Cliffs (NJ), Prentice Hall. pp. 381

Honkakorpi, J, Vihonen, J & Mattila, J. (2013). Mems sensor network based anti-sway control system for articulated hydraulic crane. Proceedings of ASME/BATH 2013 Symposium on Fluid Power and Motion Control, FPMC2013, October 6-9, Sarasota, Florida. Symposium on Fluid Power and Motion Control, pp. 1-9

Jazar, R. (2010). Theory of Applied Robotics: Kinematics, Dynamics, and Control (2nd Edition). Boston, MA, Springer Science Business Media, LLC. pp. 873

Kalmari, J., Hyyti, H. & Visala, A. (2013). Sway Estimation using Inertial Measurement Units for Cranes with a Rotating Tool. IFAC Proceedings Volumes 46. pp. 274-279

Kalmari, J., Backman, J., Visala, A. (2014). Nonlinear model predictive control of hydraulic forestry crane with automatic sway damping. Computers & Electronics in Agriculture. Vol 109. pp. 36-45

Khalil, H. (1996). Nonlinear systems (2nd Edition). Upper Saddle River, (NJ). Prentice-Hall pp. 734

Kim, Y., Hong, K., Sul, S. (2014) Anti-sway control of container cranes: Inclinometer, observer, and state feedback, International Journal of Control, Automation and Systems, Vol2(4), pp.435-449

Nakamura, Y. (1991). Advanced robotics: redundancy and optimization. Reading, MA, Addison-Wesley. pp. 337

Neupert, J., Arnold, E., Schneider, K. & Sawodny, O. (2010). Tracking and anti-sway control for boom cranes. Control Engineering Practice 18, pp. 31-44

Olsson, H., Åström, K.J., Canudas de Wit, C., Gäfvert, M. & Lischinsky, P. (1998). Friction Models and Friction Compensation, European Journal of Control, Vol. 4(3), pp. 176-195.

Ostertag, E. (2011). Mono- and Multivariable Control and Estimation: Linear, Quadratic and LMI Methods. Berlin, Heidelberg, Springer Berlin Heidelberg. p. 346

Palis, F., Palis, S. (2010), Modelling and anti-sway control of rotary cranes, Proceedings of 14th International Power Electronics and Motion Control Conference EPE-PEMC 2010, pp. T5-163 - T5-167

Siciliano, B. & Oussama, K. (2008). Springer Handbook of Robotics. In: Springer eBooks. Springer-Verlag Berlin Heidelberg. p. 1611

Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G. (2010). Robotics: modelling, planning and control. Springer-Verlag London. p. 632

Stegmann. DG 60L datasheet referred 4.8.2016, available:
http://www.portalnaukowy.edu.pl/allegro/dg60.pdf

Young, H.D., Freedman, R.A. & Sears, F.W. (2003). Sears and Zemansky's university physics: with modern physics. 11th ed. San Francisco, CA, Pearson Addison-Wesley. p. 1714

APPENDIX A: SENSORS

This appendix presents the technical data of the sensors.

Incremental encoder

All angle measurements of Hiab are made by using magnetic incremental encoders (Fraba 2014, p.12). The working principle of these encoders is based on the rotating permanent magnet, which creates a magnetic field. By measuring the changing of the magnetic field it is possible to define output pulses. When the pulse number per single revolution is known, it is possible to calculate the angle.

Figure A1 presents the Fraba IXARC incremental encoder, which is used in this crane.

Figure A6.1 *Fraba IXARC incremental encoder (Fraba 2014b)*

Table A1 presents the technical data of this sensor.

Table A1 *Technical data of Fraba (Fraba 2014b)*

Technical data	
Output	RS 422
Supply voltage	4,75-5,5 V
Pulses per revolution (ppr)	16384
Accuracy	±0,0878°
Start-up time	< 250 ms
Max rotation speed	≤ 6000 1/min
IP	68

Linear position sensor

The length of the extension boom is measured by using a linear position sensor with the incremental encoder. The incremental encoder of this sensor is Stegmann DG 60L. Table A2 presents the technical data of this sensor.

Table A2 *Technical data of the linear position sensor (Stegmann)*

Technical data	
Output	RS422
Supply voltage	4-6 V
Pulses per revolution	4096
Max length	2000 mm
Accuracy	±0.026°
Max rotation speed	6000 rpm

Pressure sensors

Two different type of pressure sensors, which made by the same manufacturer were used. First type is PTX 1400 and second is UNIX 5000 manufacturer is GE Measurement and control. Table A3 presents the technical data of these sensors (GE measurement 2014).

Table A3 *Technical data of pressure sensors*

Technical data	
Output	4-20mA
Supply voltage	7-32 V
Accuracy	±0.15% (PTX 1400)
	±0.04 (UNIX 5000)
Measurement range	0-250 bar

As shown in table A3 the output of these sensors is a current signal. For the control system this needs to be transformed to voltage. By using a 500 Ω resistor the output of these sensors is 2-10 V.

APPENDIX B: DH-PARAMETERS AND KINEMATICS EQUATIONS

This appendix presents the symbolic functions for the DH-parameters of Hiab 033 and the kinematics equations for Hiab.

```
%This file solves the forward kinematics and the differential kinemat-
ics equations for Hiab XS 033. All equations are calculated by using a
symbolic variable.

%Symbolic variables
syms A1 A2 A3 A4 alp1 alp2 alp3 alp4
syms t a alp d ox oy q1 q2 q3 q4 L1 L2 L3

%DH-parameters of the first link
a = ox;             %DH-parameter a
alp = alp1;         %DH-parameter alpha
d = oy;             %DH-parameter d
t = q1;             %DH-parameter theta

A1 = [cos(t) -sin(t)*cos(alp) sin(t)*sin(alp) a*cos(t);
    sin(t) cos(t)*cos(alp) -cos(t)*sin(alp) a*sin(t);
    0 sin(alp) cos(alp) d;
    0 0 0 1];
A1 = subs(A1,alp1,pi/2);     %Substitute value of variable alpha

%DH-parameters of the second link
a = L1;             %DH-parameter a
alp = 0;            %DH-parameter alpha
d = 0;              %DH-parameter d
t = q2;             %DH-parameter theta

A2 = [cos(t) -sin(t)*cos(alp) sin(t)*sin(alp) a*cos(t);
    sin(t) cos(t)*cos(alp) -cos(t)*sin(alp) a*sin(t);
    0 sin(alp) cos(alp) d;
    0 0 0 1];

%DH-parameters of the third link
a = 0;              %DH-parameter a
alp = alp3;         %DH-parameter alpha
d = L2;             %DH-parameter d
t = q3+pi/2;        %DH-parameter theta

A3 = [cos(t) -sin(t)*cos(alp) sin(t)*sin(alp) a*cos(t);
    sin(t) cos(t)*cos(alp) -cos(t)*sin(alp) a*sin(t);
    0 sin(alp) cos(alp) d;
    0 0 0 1];
A3 = subs(A3,alp3,-pi/2);     %Substitute value of variable alpha

%DH-parameters of the fourth link
a = 0;              %DH-parameter a
alp = alp4;         %DH-parameter alpha
d = L3+q4;          %DH-parameter d
t = 0;              %DH-parameter theta
```

```
A4 = [cos(t) -sin(t)*cos(alp) sin(t)*sin(alp) a*cos(t);
    sin(t) cos(t)*cos(alp) -cos(t)*sin(alp) a*sin(t);
    0 sin(alp) cos(alp) d;
    0 0 0 1];
A4 = subs(A4,alp4,-pi/2);     %Substitute value of variable alpha

A04 = A1*A2*A3*A4;         %Multiple matrices to each other
A04 = simplify(A04);       %Simplify matrix

%Position, Jacobian and Jacobian inverse for 4DOF system
position = [A04(1,4);A04(2,4);A04(3,4)];

%Symbolic Jacobian matrix
Jacobian = jacobian(position, [q1;q2;q3;q4]);

%Symbolic Jacobian inverse matrix
Jacobian_inv =
simplify(transpose(Jacobian)*inv((Jacobian*transpose(Jacobian))));


%Position, Jacobian and Jacobian inverse to 2DOF
%Substitute constant values in the variables of position vector
position = [A04(1,4);A04(3,4)];
position_2dof = subs(position, [q1,q4],[0,0]);

%Symbolic Jacobian matrix
Jacobian_2dof = jacobian(subs(position, [q1,q4],[0,0]),[q2,q3]);

%Jacobian inverse matrix
Jacobian_2dof_inv = inv(Jacobian_2dof);

%Position, Jacobian and Jacobian inverse for 3DOF system
%Substitute constant values of variables to position vector
position = [A04(1,4);A04(3,4)];
position_3dof = subs(position,q1,0);

%Symbolic Jacobian matrix
Jacobian_3dof = jacobian(subs(position,q1,0),[q2,q3,q4]);
Jacobian_3dof_inv = simpli-
fy(transpose(Jacobian_3dof)*inv((Jacobian_3dof*transpose(Jacobian_3dof
))));
```

APPENDIX C: CARTESIAN ACCELERATIONS

This appendix presents the symbolic functions when the crane includes 4DOF, 3DOF and 2 DOF in actuator space.

```
%This file differentiating Jacobian matrices for 4dof, 3dof and 2dof
system.
%Position vectors are solved in FK_and_Jacobian-file.

run('FK_and_Jacobian');

%Symbolic variables
syms A1 A2 A3 A4 alp1 alp2 alp3 alp4
syms t a alp d ox oy q1 q2 q3 q4 L1 L2 L3
syms q1_vel q2_vel q3_vel q4_vel
syms q1_acc q2_acc q3_acc q4_acc

%4DOF
%Velocity and acceleration vectors
q_vel = [q1_vel;q2_vel;q3_vel;q4_vel];
q_acc = [q1_acc;q2_acc;q3_acc;q4_acc];

%Symbolic solution
Jacobian_derivation_4dof =
diff(Jacobian,q1)*q1_vel+diff(Jacobian,q2)*q2_vel+...
diff(Jacobian,q3)*q3_vel+diff(Jacobian,q4)*q4_vel;

Acc_4dof = simplify(Jacobian*q_vel + Jacobian_derivation_4dof*q_acc)

%3DOF
%Velocity and acceleration vector
q_vel = [q2_vel;q3_vel;q4_vel];
q_acc = [q2_acc;q3_acc;q4_acc];

%Symbolic solution
Jacobian_derivation_3dof =
diff(Jacobian_3dof,q2)*q2_vel+diff(Jacobian_3dof,q3)*q3_vel+...
                              diff(Jacobian_3dof,q4)*q4_vel;
%Acceleration
Acc_3dof = simplify(Jacobian_3dof*q_vel +
Jacobian_derivation_3dof*q_acc)

%2DOF
%Velocity and acceleration vectors
q_vel = [q2_vel;q3_vel];
q_acc = [q2_acc;q3_acc];

%Symbolic solution
Jacobian_derivation_2dof =
diff(Jacobian_2dof,q2)*q2_vel+diff(Jacobian_2dof,q3)*q3_vel;

%Acceleration
Acc_2dof = simplify(Jacobian_2dof*q_vel +
Jacobian_derivation_2dof*q_acc)
```

APPENDIX D: JOINT TO ACTUATOR TRANSFORMATION

This appendix presents the joint-to-actuator transformation matrices.

```
%This file presents transformation matrices from the joint space to
the actuators space

syms L11 L22 L12 L21 c1 c2
syms q1 q2 q3 q4 r
syms q1_vel q2_vel q3_vel q4_vel
syms q1_acc q2_acc q3_acc q4_acc

%4DOF
%Transform matrix from the joint space to the actuator space
c_4dof = [r 0 0 0;
    0 sqrt(L11^2+L12^2-2*L11*L12*cos(q2)) 0 0;
    0 0 sqrt(L21^2+L22^2-2*L21*L22*cos(q3)) 0;
    0 0 0 1];

%Jacobian matrix
A_4dof = [r 0 0 0;
    0 (L11*L12*sin(q2))/c1 0 0;
    0 0 (L21*L22*sin(q3))/c2 0;
    0 0 0 1];

%Differentiation
c_vel = A_4dof*[q1_vel;q2_vel;q3_vel;q4_vel];

%Acceleration
A_diff_4dof = diff(A,q1)*q1_vel+diff(A,q2)*q2_vel+...
         diff(A,q3)*q3_vel+diff(A,q4)*q4_vel;

c_acc =
A_4dof*[q1_acc;q2_acc;q3_acc;q4_acc]+A_diff_4dof*[q1_vel;q2_vel;q3_vel
;q4_vel];

%3DOF
%Transform matrix from joint space to actuator space
c_3dof = [sqrt(L11^2+L12^2-2*L11*L12*cos(q2)) 0 0;
    0 sqrt(L21^2+L22^2-2*L21*L22*cos(q3)) 0;
    0 0 1];

%Jacobian matrix
A_3dof = [(L11*L12*sin(q2))/c1 0 0;
    0 (L21*L22*sin(q3))/c2 0;
    0 0 1];

%Differentiation
c_vel_3dof = A_3dof*[q2_vel;q3_vel;q4_vel];

%Acceleration
A_diff_3dof = diff(A_3dof,q2)*q2_vel+...
         diff(A_3dof,q3)*q3_vel+diff(A_3dof,q4)*q4_vel;

c_acc_3dof =
A_3dof*[q2_acc;q3_acc;q4_acc]+A_diff_3dof*[q2_vel;q3_vel;q4_vel];
```

```
%2DOF
%Transform matrix from the joint space to the actuator space
c_2dof = [sqrt(L11^2+L12^2-2*L11*L12*cos(q2)) 0;
         0 sqrt(L21^2+L22^2-2*L21*L22*cos(q3))];
%Jacobian matrix
A_2dof = [(L11*L12*sin(q2))/c1 0;
          0 (L21*L22*sin(q3))/c2];

%Differentiation
c_vel_2dof = A_2dof*[q2_vel;q3_vel];

%Acceleration
A_diff_2dof = diff(A_2dof,q2)*q2_vel+...
         diff(A_2dof,q3)*q3_vel;

c_acc_2dof = A_2dof*[q2_acc;q3_acc]+A_diff_2dof*[q2_vel;q3_vel];
```

APPENDIX E: ANGULAR ACCELERATIONS

This appendix presents the symbolic solutions of angular accelerations of the tool.

```
%This file solves equations for the angular accelerations of the tool

%Symbolic variables
syms l1 l2 m g Ia Ib            %Constant values
syms dx dy dz da db             %Linear velocity and angular velocity
variables
syms a(t) b(t) x(t) y(t) z(t)    %Position parameters which depends on
time
syms dx(t) dy(t) dz(t) da(t) db(t) %Velocities which depends on time
syms b_var a_var x_var y_var z_var x_vel y_vel z_vel a_vel b_vel  %Ve-
locity and position parameters which not depens on time
syms x_acc y_acc z_acc a_acc b_acc  %Linear and angular accelerations
syms bv av      %Friction variables

%The center of mass position in the tool coordinate frame
l3 = l1+l2*cos(b);

xc = -l3*cos(a);        %The center of mass x-coordinate
yc = -l3*sin(a);        %The center of mass y-coordinate
zc = -l2*sin(b);        %The center of mass z-coordinate

%The center of mass position in the world coordinate frame
xw = x - yc;            %The center of mass x-coordinate
yw = y - zc;            %The center of mass y-coordinate
zw = z + xc;            %The center of mass z-coordinate

xw_vel = diff(xw,t);    %The velocity of the center of mass x-
coordinate
yw_vel = diff(yw,t);    %The velocity of the center of mass y-
coordinate
zw_vel = diff(zw,t);    %The velocity of the center of mass z-
coordinate

%The kinetic energy of the tool
T =
(1/2)*m*(xw_vel^2+yw_vel^2+zw_vel^2)+(1/2)*Ia*diff(a,t)^2+(1/2)*Ib*dif
f(b,t)^2;

%The potential energy of the tool
V = m*g*zw;

%Lagrangian equation
L = T-V;

%Next need to replaced differentiations with variables which doesn't
depend on time because the symbolic toolbox cannot do diffentiation to
the symbolic functions

L = subs(L,diff(x,t),x_vel);
L = subs(L,diff(y,t),y_vel);
L = subs(L,diff(z,t),z_vel);
L = subs(L,diff(a,t),a_vel);
```

```
L = subs(L,diff(b,t),b_vel);
L = subs(L,x(t),x_var);
L = subs(L,y(t),y_var);
L = subs(L,z(t),z_var);
L = subs(L,a(t),a_var);
L = subs(L,b(t),b_var);

%Partial differentiation to joint 1
derivation1 = simplify(diff(L,a_vel));        %Part of Lagrangian
equation
derivation2 = simplify(diff(L,a_var));        %Part of Lagrangian
equation

%Substitute variables which depending on time to the equations
derivation1 =
subs(derivation1,[x_vel,y_vel,z_vel,a_vel,b_vel,a_var,b_var],...
    [dx(t),dy(t),dz(t),da(t),db(t),a(t),b(t)]);
derivation2 =
subs(derivation2,[x_vel,y_vel,z_vel,a_vel,b_vel,a_var,b_var],...
    [dx(t),dy(t),dz(t),da(t),db(t),a(t),b(t)]);

%The moment of joint 1 by using Lagrangian formulation
moment_a = diff(derivation1,t)-derivation2;

%Substitute variables to the differentiations
moment_a = subs(moment_a,diff(dx,t),x_acc);
moment_a = subs(moment_a,diff(dy,t),y_acc);
moment_a = subs(moment_a,diff(dz,t),z_acc);
moment_a = subs(moment_a,diff(da,t),a_acc);
moment_a = subs(moment_a,diff(db,t),b_acc);
moment_a = subs(moment_a,diff(a,t),da(t));
moment_a = subs(moment_a,diff(b,t),db(t));
moment_a = subs(moment_a,a_var,a(t));
moment_a = subs(moment_a,b_var,b(t));
moment_a = simplify(moment_a);           %Simplify equation

%Partial derivation for the joint 2
derivation1 = simplify(diff(L,b_vel));        %Part of Lagrangian
equation
derivation2 = simplify(diff(L,b_var));        %Part of Lagrangian
equation

%Substitute variables which depending on time to the equations
derivation1 =
subs(derivation1,[x_vel,y_vel,z_vel,a_vel,b_vel,a_var,b_var],...
    [dx(t),dy(t),dz(t),da(t),db(t),a(t),b(t)]);
derivation2 =
subs(derivation2,[x_vel,y_vel,z_vel,a_vel,b_vel,a_var,b_var],...
    [dx(t),dy(t),dz(t),da(t),db(t),a(t),b(t)]);

%Calculate moment for joint 2 by using Lagrangian formulation
moment_b = diff(derivation1,t)-derivation2;

%Substitute variables to the differentiations
moment_b = subs(moment_b,diff(dx,t),x_acc);
moment_b = subs(moment_b,diff(dy,t),y_acc);
moment_b = subs(moment_b,diff(dz,t),z_acc);
moment_b = subs(moment_b,diff(db,t),b_acc);
moment_b = subs(moment_b,diff(da,t),a_acc);
moment_b = subs(moment_b,diff(a,t),da(t));
```

```
moment_b = subs(moment_b,diff(b,t),db(t));
moment_b = subs(moment_b,a_var,a(t));
moment_b = subs(moment_b,b_var,b(t));
moment_b = simplify(moment_b);       %Simplify equations

%Solve equations for the angular accelerations by using moment equa-
tions. In this case there is used viscose friction model

a_acceleration = solve(moment_a == -m*av*da(t),a_acc);
b_acceleration = solve(moment_b == -m*bv*db(t),b_acc);
```

APPENDIX F: LINEAR DYNAMIC MODEL

This appendix presents the linearization of the dynamic equations.

```
%This file linearizes the angular acceleration equations and
%the linear acceleration equations.

%Symbolic variables
syms x1 x2 x3 x4 x5 x6 x7 x8 x9 x10   %State variables
syms u1 u2 u3                         %Control variables
syms x_cm y_cm z_cm                   %Position of the center of mass

%Accelerations of the center of mass
xw_acc = diff(xw_vel,t);    %Acceleration of center of mass x-
coordinate
yw_acc = diff(yw_vel,t);    %Acceleration of center of mass y-
coordinate
zw_acc = diff(zw_vel,t);    %Acceleration of center of mass z-
coordinate

%Substitute state variables to the equations
a_acceleration =
subs(a_acceleration,[x_acc,y_acc,z_acc,a(t),da(t),b(t),db(t)],...
    [u1,u2,u3,x7,x8,x9,x10]);
b_acceleration =
subs(b_acceleration,[x_acc,y_acc,z_acc,a(t),da(t),b(t),db(t)],...
    [u1,u2,u3,x7,x8,x9,x10]);

%Substitute the state variables and the angular accelerations to the
equations
xw_acc = subs(xw_acc,diff(diff(x,t),t),u1);
xw_acc = subs(xw_acc,diff(diff(y,t),t),u2);
xw_acc = subs(xw_acc,diff(diff(z,t),t),u3);
xw_acc = subs(xw_acc,diff(diff(a,t),t),a_acceleration);
xw_acc = subs(xw_acc,diff(diff(b,t),t),b_acceleration);
xw_acc = subs(xw_acc,diff(a,t),x8);
xw_acc = subs(xw_acc,diff(b,t),x10);
xw_acc = subs(xw_acc,a(t),x7);
xw_acc = simplify(subs(xw_acc,b(t),x9));

yw_acc = subs(yw_acc,diff(diff(x,t),t),u1);
yw_acc = subs(yw_acc,diff(diff(y,t),t),u2);
yw_acc = subs(yw_acc,diff(diff(z,t),t),u3);
yw_acc = subs(yw_acc,diff(diff(a,t),t),a_acceleration);
yw_acc = subs(yw_acc,diff(diff(b,t),t),b_acceleration);
yw_acc = subs(yw_acc,diff(a,t),x8);
yw_acc = subs(yw_acc,diff(b,t),x10);
yw_acc = subs(yw_acc,a(t),x7);
yw_acc = simplify(subs(yw_acc,b(t),x9));

zw_acc = subs(zw_acc,diff(diff(x,t),t),u1);
zw_acc = subs(zw_acc,diff(diff(y,t),t),u2);
zw_acc = subs(zw_acc,diff(diff(z,t),t),u3);
zw_acc = subs(zw_acc,diff(diff(a,t),t),a_acceleration);
zw_acc = subs(zw_acc,diff(diff(b,t),t),b_acceleration);
zw_acc = subs(zw_acc,diff(a,t),x8);
zw_acc = subs(zw_acc,diff(b,t),x10);
zw_acc = subs(zw_acc,a(t),x7);
```

```
zw_acc = simplify(subs(zw_acc,b(t),x9));

%Nonlinear state-space derivations are
%x1 = x2
%x2 = xcm_acc
%x3 = x4
%x4 = ycm_acc
%x5 = x6
%x6 = zcm_acc
%x7 = x8
%x8 = a_acceleration
%x9 = x10
%x10 = b_acceleration

%Controls
%u1 = x_acc
%u2 = y_acc
%u3 = z_acc

%Linear matrices by using Jacobian matrix
A = jacobian(
[x2,xw_acc,x4,yw_acc,x6,zw_acc,x8,a_acceleration,x10,b_acceleration],
[x1,x2,x3,x4,x5,x6,x7,x8,x9,x10]);

B = jacobian(
[x2,xw_acc,x4,yw_acc,x6,zw_acc,x8,a_acceleration,x10,b_acceleration],
[u1,u2,u3]);

%C matrix
C = [1 0 0 0 0 0 0 0 0 0;
     0 0 1 0 0 0 0 0 0 0;
     0 0 0 0 1 0 0 0 0 0;
     0 0 0 0 0 0 1 0 0 0;
     0 0 0 0 0 0 0 0 1 0];

%Substitute the equilibrium point and the constant values to the con-
trol matrix and the system matrix

A = subs(A,[x2,x4,x6,x7,x8,x9,x10,u1,u2,u3],[0,0,0,0,0,0,0,0,0,0]);
B = subs(B,[x2,x4,x6,x7,x8,x9,x10,u1,u2,u3],[0,0,0,0,0,0,0,0,0,0]);
```

APPENDIX G: LQR AND LQG TUNING

This appendix presents tuning commands to the state feedback controller and the estimator.

```
%This file calculates optimal gains to the state feedback and the es-
timator

%Sample time
Sampletime = 0.002;

%Covariance matrices for the measurements and the outputs
W = eye(10)*0.001;
V = eye(5)*0.01;

%Weight matrices
r = 8;          %Tuning gain of the Bryson rule

%Maximum deviations
qr = [1/0.05^2   0     0     0    0
      0     1/0.1^2    0     0    0
      0     0     1/0.06^2     0   0
      0     0     0     1/0.15^2   0
      0     0     0     0     1/0.15^2];

%Weighting matrix Q
Q = r*C'*qr*C;

%Weighting matrix R
R = [1/0.15^2     0     0
     0     1/0.15^2     0
     0     0     1/0.2^2];

c = c2d(ss(A,B,C,0),Sampletime);         %Discrete system

%Optimal gain for the state feedback controller
[K,S,E] = dlqr(c.a,c.b,Q,R,0);

%Optimal gain to Kalman filter
sys = ss(c.a,[c.b eye(10)],c.c,0,Sampletime);

[Kest,M,P,L,Z] = kalman(sys,W,V);

%The reference matrix for the state feedback controller
Ch = [1 0 0 0 0 0 0 0 0 0;
      0 0 1 0 0 0 0 0 0 0;
      0 0 0 0 1 0 0 0 0 0];

N = inv([c.a-eye(10) c.b;Ch zeros(3)])*[Ch'*0;eye(3)];
Nx =
[N(1,:);N(2,:);N(3,:);N(4,:);N(5,:);N(6,:);N(7,:);N(8,:);N(9,:);N(10,:
)];
Nu = [N(11,:);N(12,:);N(13,:)];

Nall = Nu+K*Nx;
```

APPENDIX H: THE REFERENCE PATH

This appendix presents the working principle of the reference path generator.

In this thesis the reference paths of the Cartesian position of the boom tip haves been determined by using a five-degree polynomial (Jazar p.736) which is of form

$$p(t) = h_0 + h_1 t + h_2 t^2 + h_3 t^3 + h_4 t^4 + h_5 t^5 \tag{H1}$$

where *t* is time and *h* is gain. This polynomial can satisfy six conditions of the reference path which are

$$\begin{array}{lll} p(t_0) = \mathrm{p}_0 & \dot{p}(t_0) = p_{0\_vel} & \ddot{p}(t_0) = p_{0\_acc} \\ p(t_f) = p_f & \dot{p}(t_f) = p_{f\_vel} & \ddot{p}(t_f) = p_{f\_acc} \end{array} \tag{H2}$$

where p_0 is the start position and p_f is the final position.

As equation (H2) shows it is possible to give start and final values to position, velocity and acceleration. The polynomial (H1) describes the path between two points. The full reference path can be defined by combining paths between different points together. The gains of polynomial (H1) can be determined by solving the equation (Jazar p.736)

$$\begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 & t_0^4 & t_0^5 \\ 0 & 1 & 2t_0 & 3t_0^2 & 4t_0^3 & 5t_0^4 \\ 0 & 0 & 2 & 6t_0 & 12t_0^2 & 20t_0^3 \\ 1 & t_f & t_f^2 & t_f^3 & t_f^4 & t_f^5 \\ 0 & 1 & 2t_f & 3t_f^2 & 4t_f^3 & 5t_f^4 \\ 0 & 0 & 2 & 6t_f & 12t_f^2 & 20t_f^3 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix} = \begin{bmatrix} p_0 \\ p_{0\_vel} \\ p_{0\_acc} \\ p_f \\ p_{f\_vel} \\ p_{f\_acc} \end{bmatrix}. \tag{H3}$$

For example, for the conditions

$$\begin{array}{lll} p(0) = 2 & \dot{p}(0) = 0 & \ddot{p}(0) = 0 \\ p(0) = 8 & \dot{p}(2) = 0 & \ddot{p}(2) = 0 \end{array} \tag{H4}$$

the reference signals waveforms for the position, velocity and accelerations is presented in Figure H1.

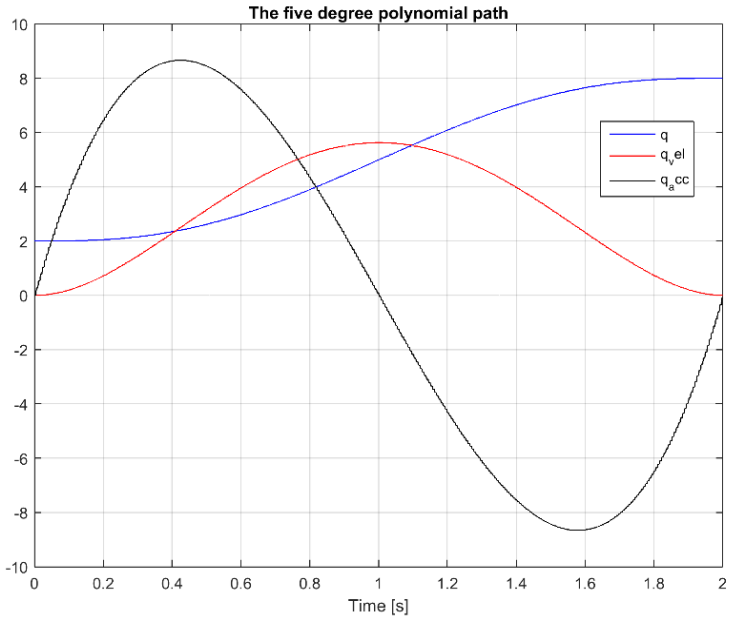

Figure H1 *The polynomial path*

Figure H1 shows that reference signals fulfil conditions.